DSTO

Navy Specification Study
Report 3
Requirements and Specifications

Andrew P. Gabb and
Derek E. Henderson

Approved for Public Release

DEPARTMENT ◆ OF DEFENCE
DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION

Navy Specification Study
Report 3
Requirements and Specifications

Andrew P. Gabb and Derek E. Henderson

Information Technology Division
Electronics and Surveillance Research Laboratory

DSTO-TR-0192

ABSTRACT

Technical Report

This report is one of a series examining the specification of complex computer based systems in the Royal Australian Navy (RAN). The study was carried out under Task NAV 93/067, *Review of Specification and Evaluation Practices.* This report analyses the needs of specifications and makes recommendations for their improvement.

DEPARTMENT OF DEFENCE

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION

Published by

DSTO Electronics and Surveillance Research Laboratory
PO Box 1500
Salisbury South Australia 5108

Telephone: (08) 259 5296
Fax: (08) 259 5980

AR-~~008-3112~~ 009-312
September 1995

APPROVED FOR PUBLIC RELEASE

Authors

Andrew P. Gabb
Information Technology Division

Andrew Gabb *has been with the Australian Defence Science and Technology Organisation since graduating in Electrical Engineering and Computing Science at the University of Adelaide in 1971. In that time he has acted as a consultant to numerous major projects in the Australian Department of Defence, in the areas of systems and software engineering primarily for combat systems. He is currently a Principal Research Scientist conducting research into C3I front end systems engineering and procurement methods.*

Derek E. Henderson
Information Technology Division

Derek E. Henderson *has been with the Australian Defence Science and Technology Organisation (DSTO) since 1991. Prior to that he was with the Ministry of Defence (MOD) in the UK after graduating in Electronic Systems Engineering at the Royal Military College of Science, Shrivenham, UK. During his time with the MOD he was a member of a project team managing the procurement of software for a submarine command system. Since joining DSTO he has undertaken studies for the Australian Department of Defence in the areas of specification, interoperability, and acquisition of information systems. His current research interests include C3I front end systems engineering, evolutionary acquisition and C3I database systems.*

THIS PAGE INTENTIONALLY BLANK

Contents

Abbreviations

AII	Australian Industry Involvement
AINS	Acceptance into Naval Service
BITE	Built In Test Equipment
CASE	Computer Aided Software Engineering
CCTA	Government Centre for Information Systems (UK)
CEP	Circular Error Probable
CEPMAN	The Capital Equipment Procurement Manual
CORE	Controlled Requirements Expression
COTS	Commercial off the shelf
DI	Developmental Item
DOR	Detailed Operational Requirements
DSTO	Defence Science and Technology Organisation
JSD	Jackson System Development
NDI	Non-developmental Item
ODBC	Open Database Connectivity
OLE	Object Linking and Embedding
OOA	Object Oriented Analysis
OORA	Object Oriented Requirements Analysis
OPEVAL	Operational Evaluation
PC	Personal Computer (IBM compatible)
RAN	Royal Australian Navy
RFP	Request for Proposal
RFT	Request for Tender
SSADM	Structured Systems Analysis and Design Methodology
TLR	Top Level Requirements
TLS	Top Level Specification

THIS PAGE INTENTIONALLY BLANK

1. Introduction

1.1 General

This report is one of a series examining the specification of complex computer based systems in the Royal Australian Navy (RAN). The study was carried out under Task NAV 93/067, *Review of Specification and Evaluation Practices*. This report analyses the needs of specifications and makes recommendations for their improvement.

The full series is as follows:

Report 1: Industry survey	A survey of industry perceptions of Navy specifications.
Report 2: Current policy and practice	Examines the current policy and practice regarding the development of specifications in Navy. (Not Public Release.)
Report 3: Requirements and specifications	A comprehensive review of the needs for Navy specifications, providing recommendations for their improvement.
Report 4: Executive summary and final recommendations	Consolidated recommendations arising from Reports 1, 2 and 3. (Not Public Release.)

Although carried out under a DSTO task endorsed by Navy, this study is essentially an external review of the relevant practices within Navy. In providing recommendations, no attempt has been made to assess the impact of their implementation on the organisational structure of Navy Materiel Division or related activities within Navy, or the feasibility of their implementation.

1.2 Scope

The specifications under review are those provided as part of a Request for Proposal (RFP) or Request for Tender (RFT). In general, these specifications contain performance and functional requirements as well as non-functional requirements. A traditional "big bang" project is also assumed, in which all the performance and functionality needs to be specified prior to the contract being let.

Typically, computer based systems are difficult to specify because of the large numbers of functions which are required and the high level of interaction between different functions. Operational computer based systems include combat systems, information systems and ship control and management systems. Many of these systems have critical real-time requirements, and a very large software component (some Navy combat systems include millions of

lines of code). These factors increase the difficulty of development and hence increase the importance of getting the initial requirements correct.

While this paper discusses the improvement of Navy specifications, we have seen no reason to believe that Navy specifications are inferior to other Defence specifications. Several participants in our industry survey (Report 1 in this series) suggested that Navy specifications are often superior to other specifications, particularly in providing a clear and reasonably high level representation of the requirements. In many ways, this paper provides suggestions which are relevant to all Defence specifications for operational systems.

1.3 Importance of specifications

It is now widely accepted that the majority of problems in the development of complex systems stem from incorrect or inadequate requirement specifications. Studies of large projects have also shown that more than half of the system errors are introduced in the requirements definition phase [Black 1989; Davis 1993a].

Such errors are difficult to detect. Some are revealed during the design process when it becomes clear that the emerging design, although compliant with the system specification in the contract Statement of Work, will not satisfy the users' needs. Others are detected during acceptance testing or operational evaluation (OPEVAL). Still more are found during operation of the system in service.

In addition, the errors can be very costly to fix, not only because they can affect the architectural design of the system, but also because they are found so late in the development life cycle. Table 1 [Davis 1993a] shows the relative costs of correction of errors at different stages in system development and use.

Table 1. *Cost to repair software at different life cycle stages*

Stage	Relative cost of repair
Requirements	0.1-0.2
Design	0.5
Coding	1
Unit test	2
Acceptance test	5
Maintenance	20

The fact that the systems under review are complex, have a large software component, and are often innovative, increases the risk [Kirkpatrick et al. 1992] and emphasises the need to concentrate on improving the specifications. This was also shown in an analysis of the lessons learned in the ANZAC Ship project where one of the conclusions was that the effort spent in getting the specification right would pay significant dividends to Navy.

Unfortunately, the capture and definition of requirements is difficult, and is a task that few staff can perform well. When coupled with the potential for serious adverse impacts to system performance, and to project cost and schedule, it is evident that this problem demands immediate and thorough treatment.

1.4 Industry survey

As part of this study, we undertook a survey of Australian industry perceptions of Navy's specifications. The results of this survey are recorded in Report 1 in this series.

Problems identified with Navy specifications included the following. Although no specification was singled out as having all these faults, all specifications were perceived as having some.

- Requirements which are difficult to understand, and which appeared to be based on assumed knowledge.
- Inappropriate copying of requirements from other specifications or glossy brochures.
- Requirements which prevent the use of technological advantages.
- Over-embracing requirements (excessive scope).
- Inadequate requirements for environmental conditions, usability and shore based facilities.
- Lack of evident traceability from operational to functional requirements.
- Inappropriate or ambiguous specification of standards.
- Uneven level of specification including overspecification and underspecification.
- Vague requirements.
- Inconsistency of formatting and structure.
- Merging of several requirements into a single clause.
- Scattering of related requirements throughout the specification and the RFT package.
- Uncertain precedence and priority of requirements.
- Insufficient definition of the terms used.
- Poor specification of non-functional requirements.

Additional recommendations for improvement included:

- In classified documents, the provision of portion marking or separation of classified/unclassified requirements.
- Provision of specifications in electronic form in a popular industry standard format.
- Disciplined and standardised use of tools for specification preparation.
- Pre-release of draft specifications to potential tenderers.
- Additional access of tenderers to information regarding the proposed operational use of the system.
- Formal review of specifications against comprehensive guidelines.

Several of these findings were also revealed in the *Costs of Tendering Industry Survey* [1994].

1.5 Current policy and practice

We also surveyed the current policy and practice relating to specifications in Navy and Defence. The results of this survey are recorded in Report 2 in this series.

The survey showed that policy is defined for the development of specifications (mainly in *The Capital Equipment Procurement Manual* [CEPMAN 1992]) and it is generally followed. However the guidance, while sound, is at a high level, and is generally inadequate as a basis for specification development. Other guidance is provided in individual handbooks. The guidance in these is patchy and incomplete and does not address many of the issues discussed in this paper.

1.6 Layout of this report

Section 2	Discusses the terminology of requirements and specifications, including the different meanings sometimes ascribed to different terms.
Section 3	Considers what kind of specifications are best for Navy. In doing so it addresses the following questions: What are the actual needs of specifications? Why they are written and who they are written for? How do performance based specifications differ from those which are currently used? What are the alternatives to natural language specifications?
Section 4	Looks at requirements and requirements engineering. Requirements capture, analysis, definition and validation are discussed and it is shown why they are important. The relationship between specifications and evaluations is also examined.
Section 5	Defines a checklist of the attributes needed by good specifications.
Section 6	Provides a detailed discussion of issues which must be considered in the development of specifications.
Section 7	Discusses ways in which communication with tenderers may enhance the quality and value of specifications.
Section 8	Considers several diverse issues including personnel issues, the use of tools in specification development, specifying for existing components (COTS and NDIs), the effects of requirements for a high level of integration, Australian industry involvement and alternative acquisition strategies.
Section 9	Examines the need for a systematic process for specification development, including evaluation of the quality of specifications.
Sections 10 and 11	Summarise the report with recommendations and conclusions.

2. Terminology

This section defines the terminology used in this paper. Many of the terms used in systems engineering in general, and requirements engineering in particular, are also commonly used in other fields, and hence many of the terms are used differently by different writers. Because of the potential for misinterpretation, and the fact that many of the definitions depend on other words which also have a specialised use, we have organised this section in the form of a discussion rather than an alphabetically arranged list of definitions. In addition, this allows discussion of the meanings as well as providing the definitions.

Many of the definitions in this section are based on those in *System and Software Requirements Engineering* [Thayer and Dorfman 1990] and the draft standard *Systems Engineering* [draft MIL-STD-499B 1994]. Where there is a variety of meanings in current use we have attempted to choose the meaning which appears to be most popular in practical (as opposed to theoretical) texts.

System

A **system** is a collection of hardware, software, documentation, people, facilities and procedures organised to accomplish some common objective. Typical complex computer based systems include combat systems, ship control systems and information systems.

Users

Users of a system are the operators and supporters of the system, and the trainers that train the operations and support personnel. This definition (from MIL-STD-499B) extends the traditional meaning of users to include the support personnel, including trainers. It recognises the fact that the needs of the support personnel need to be taken into account in the requirements for an operational system.

Requirement

Genuine requirement

Written requirement

A **requirement** is a capability needed by a user to solve a problem or achieve an objective. When expressed in words or other notations, usually in a specification, it is also the statement or definition of that capability. It is important to recognise that the word "requirement" has both these meanings. Whenever it is necessary to distinguish between the two we will use the terms **genuine requirement** and **written requirement**. Because of the limitations and ambiguities in the use of language, the genuine and written requirements will rarely be exactly identical. This is not a resolvable problem (although many have tried); ideas and concepts are usually impossible to represent completely and accurately in words, or in any other notation.

Derived requirement

Explicit requirement

Implicit requirement

A **derived requirement** is a requirement which is determined to be necessary for a higher level requirement to be met. The higher level requirement is typically either in the same or a higher level specification. For example, if there is a high level requirement for a printed records of events, derived requirements may define the need for the collection and printing of those records. Requirements may also be explicit or implicit. An **explicit requirement** is one which is defined in the specification. An **implicit requirement** is one which is implied by other requirements. In the above example, there is an implicit requirement for some form of printer.

Functional requirement

Performance requirement

Behavioural requirement

A **functional requirement** is one which describes some aspect of the required system behaviour in terms of what function is required, i.e. *what* it should do. A **performance requirement** expands upon a (sometimes implicit) functional requirement, usually indicating how well that function will be performed, i.e. *how well* it should do it. Performance requirements are usually quantitative, specifying for example speed, response times, accuracy or probability of detection. Together, functional and performance requirements are sometimes referred to as **behavioural requirements** .

The functional requirements of a radio, for example, may include the need to receive, transmit, and to provide facilities to change the operational frequency. The performance requirements may include the need to receive and transmit within specific frequency ranges and also the frequency discrimination. Consider however the requirement for a radio to operate on 4 discrete nominated frequencies. Strictly, this is a functional requirement because it specifies what the radio does. But it may also be regarded as a performance requirement, partly because it includes numbers (the frequencies) and partly because it specifies a range. This example illustrates the fine line between functional and performance requirements. In most cases they can be treated identically.

Some texts define non-functional requirements to include performance (as opposed to functional) requirements. Some older texts address functional and performance requirements simply as "requirements" and non-functional requirements as "constraints". This is a rather curious distinction because almost all requirements constrain the design or selection of equipment in some way.

Specification

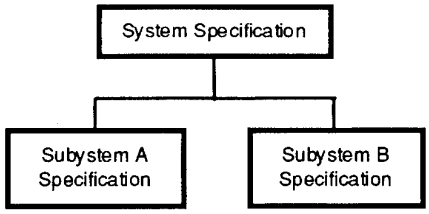
Specification tree

Specification level

A **specification** is a document that prescribes, in a complete precise verifiable manner, the requirements, design, behaviour, or other characteristics of a system or system component. In this paper we will generally view a specification as the collected requirements for the system, although it may contain other, usually informative, material. A **specification tree** is a group of related specifications with an established hierarchy and precedence. Figure 1 shows a simple specification tree for a system with two major subsystems. The system specification will normally take precedence over the subsystem specifications. **Specification level** is based on the position of the specification in the tree.

A high level specification is typically high in the tree and contains less detailed requirements than those in a lower level specification.

***Figure 1**. Simple specification tree*



Operational requirements specification

User requirements specification

An **operational requirements specification** contains requirements which relate directly to the users' needs in carrying out their mission. It is usually written in the users' language and describes the mission or missions, the operations that need to be performed and the capabilities needed to perform them. It is often the highest level specification in the specification tree. This is also sometimes called a **user requirements specification**. The Top Level Requirement and Detailed Operational Requirement discussed in section 3.2 are examples of operational requirements specifications.

Functional specification

Performance specification

RFT specification

A **functional** or **performance specification** (the terms tend to be used interchangeably) contains functional, performance and non-functional requirements and is derived from the operational specification. It is limited to describing the requirements for the system (or subsystem) and only those non-functional requirements which are considered essential. It does not assume or specify particular solutions, and is often used as the basis for system design. In this paper the term **RFT specification** is used to mean a functional and performance specification used as a basis for Request for Tender. The Top Level Specification discussed in section 3.2, and the System/Segment Specification defined in MIL-STD-490A are examples of functional or performance specifications.

A-spec

System/ Segment Specification

Segment

The frequent use of MIL-STD-490A (*Specification Practices*) has resulted in terminology which is widely used and often misunderstood. Discussion of MIL-STD-490A specifications is included here only for completeness.

- An **A-spec** or **System/Segment Specification** "states the technical and mission requirements for a system/segment as an entity, allocates requirements to functional areas, documents design constraints, and defines the interfaces between or among the functional areas". The "system" in this case refers to everything which is to be provided under the contract, e.g. a ship, its subsystems and support systems. A **segment** is a convenient part of a system which is considered more complex than a prime item (see below). Thus the A-spec format may be used to specify a total "system" or a segment of that system. A System Specification is usually the highest level specification used in a contract.

B-spec

- A **B-spec** is a development specification. "Development specifications state the requirements for the design or engineering development of a product during the development period. Each development specification shall be in sufficient detail to describe effectively the performance characteristics that each configuration item is to achieve when a developed configuration item is to evolve into a detail design for production." There are several types of development specification including:

Prime Item Development Specification

  - **Prime Item Development Specification** (B1). "Applicable to a complex item such as an aircraft, missile, launcher equipment, fire control equipment, radar set, training equipment, etc."

Critical Item Development Specification

- **Critical Item Development Specification** (B2). "Applicable to a configuration item which is below the level of complexity of a prime item but which is engineering critical or logistics critical." In this case the meaning of "critical" is misleading and is derived from the needs for configuration management and separate testing. In practice, most deliverable system components are included as critical items or parts thereof.

Facility or Ship Development Specification

- **Facility or Ship Development Specification** (B4).

Software Development Specification

- **Software Development Specification** (B5). Software Requirements Specifications and Interface Requirements Specifications as defined in DOD-STD-2167A are B5 specifications.

C-spec

- A **C-spec** is a product specification, which is "applicable to any configuration item below the system level, and may be oriented toward procurement of a product through specification of primarily functional (performance) requirements or primarily fabrication (detailed design) requirements". C-specs may be used for both prime items and critical items.
- Less widely used within Navy are D-specs (process specifications) and E-specs (material specifications).

Verification

Verification is determination that a system development product meets the requirements directly imposed on that product. For example, a design specification or test specification for a system component will normally be verified against the requirements specification/s affecting that component. The component itself will be verified against its design specification. There are many products produced during system development, including requirements and design specifications, test plans and procedures, hardware and software components, and the system itself. Some are deliverable and some are not. Most products will be subject to verification; some will be subject to validation (see below). Testing is but one form of verification; other methods include inspection, demonstration, analysis and comparison with other systems. Verification can be a somewhat mechanical activity, although it needs good judgment and a thorough understanding of the development methods used.

Validation

Validation is determination that a system development product meets the users' needs. Theoretically this is simply a matter of checking that the product complies with the higher level (user requirement) specifications. In practice, the situation is more complex. The definition of validation deliberately refers to "users' needs". As stated previously, there will always be a difference between the written requirement (the user specification) and the genuine requirement (the users' needs). Examination of a design

product may reveal deficiencies in the user requirement specifications, for example, or show that the designer has misinterpreted the written requirements. Validation tends to be a subjective activity, requiring good judgment and a sound understanding of the user requirement.

As an example, acceptance testing is a verification activity whereas operational evaluation leading to Acceptance into Naval Service (AINS) is a validation activity.

Qualification

Qualification includes both verification and validation and is the formal process of ensuring that a system product meets its overall requirements.

Requirements engineering

Requirements engineering is a systems engineering discipline concerned with defining and managing requirements. Many different terms are used for the different activities or phases involved in defining requirements, often interchangeably. The following terms are used in this paper:

Requirements capture

Requirements elicitation

- **Requirements capture** is the activity of eliciting the requirements which encapsulate the users' needs, usually from the users. This is also called **requirements elicitation**.

Requirements analysis

- **Requirements analysis** is the study of the captured users' needs to derive a coherent, complete, consistent and feasible set of requirements. This often includes modelling and tradeoff studies.

Requirements definition

- **Requirements definition** involves representing the requirements in a usable form, usually by drafting a specification.

Requirements validation

- **Requirements validation** is the process by which the requirements engineer ensures that the written requirements are an adequate expression of the genuine requirements.

Requirements tracing

- **Requirements tracing** is an activity which attempts to ensure that:
  - Every new requirement (at any level) is justified and validated, and
  - Every other requirement and product feature in the specification tree is derived from and traceable to a higher level requirement, and vice versa.

  This should ensure that any requirement can be tracked to the product feature/s which satisfy that requirement. It should also ensure that each product feature can be justified by the requirement/s which caused it to be incorporated in the system. Requirements traceability is essential for efficient verification.

Requirements engineering is essentially an iterative and interactive process. None of the above activities will occur in isolation. Each phase will involve some elements of the other

phases. For example, thorough analysis of the requirements will invariably reveal deficiencies (typically omissions or inconsistencies) from the capture phase, which will need further discussions with the users. Validation wili also reveal problems, which will necessitate further capture, analysis and definition activities.

Developmental item (DI)

Non-developmental item (NDI)

Commercial off the shelf (COTS)

A **developmental item (DI)** is a component of a system which requires development or modification to be used in the system. A **non-developmental item (NDI)** is a component of a system which does not require design and development or modification. A **commercial off the shelf (COTS)** component is a component which is widely used in commercial applications. A non-developmental item differs from a COTS item only in the fact that an NDI might only have been used in (possibly relatively few) military applications or that it is not widely used. There is often an assumption (see section 6.9.3) that COTS components have an advantage in cost and proven performance over other components.

3. Specification needs

This section examines the needs of Navy specifications with a view to determining an appropriate specification method. Alternative specification methods are examined including the use of performance based specifications. Finally a specification method is recommended.

3.1 Specifications are products

To clearly understand the needs of specifications, it is necessary to view them as products. Like a hardware or software product, a specification has users. The users' needs should be identified and analysed, the specification should be systematically designed and developed (written), and the product should be tested according to the needs of its users.

Hardware and software products sometimes fail. They fail when they are being tested, and they fail in service. We do not tend to think of specifications failing (although they obviously do, because they are wrong, or ambiguous or incomplete). Instead, we see the resultant hardware and software fail, so that it seems natural to blame the maker of that product, rather than its specification or one of its ancestors. And it is often the users of the product, the operators of the system, who are penalised. They have to accommodate or work around the area of failure, and they are neither as effective or as efficient in their tasks as they should be.

Our intention in this study is to treat specifications as products:

- Products which have users - we call them "audiences".
- Products which have developers - users, requirements engineers, analysts, specification drafters (or writers).

- Products which are tested - we call it validation of the specification.
- Products which are used - to form the technical basis for the development of a complex system.

Most importantly we stress the importance, the criticality, of the product. If our product is inadequate, the project may incur severe penalties in performance, cost and schedule (see section 1.3).

3.2 Navy specifications in context

Current Navy policy [FD(Sea 1993; FD(Sea) 1994] defines a specification hierarchy for evolving the capability requirements in a major equipment acquisition.

a. **Top Level Requirements (TLR).** The TLR is a broad performance based statement of requirement that summarises the needed capabilities.

b. **Detailed Operational Requirements (DOR).** The DOR is a detailed performance based statement of requirement, combining both platform and combat system requirements, derived directly from the TLR. At contract signature, the DOR configuration will be frozen and will be used as the basis for Acceptance into Naval Service (AINS).

c. **Top Level Specification (TLS).** The TLS is an "engineering specification to be used in the tendering and contract process". The TLS will be derived from the DOR. It is primarily these types of specification which are being reviewed in this study.

d. **Contractual Specifications.** These specifications are included in the contract Statement of Work and define the functional baseline for the contract. They are based on the successful tender and the TLS and are agreed during contract negotiation. Such specifications usually contain a mixture of requirements, products and product development requirements.

It can be seen that in developing each of these specifications, their quality and effectiveness depends critically on the specification above them in the hierarchy. Errors in the DOR for example are likely to propagate into the TLS and the contract specifications.

Although this study is primarily aimed at RFT specifications (the Top Level Specifications) most of the findings apply equally to the Top Level Requirements and the Detailed Operational Requirements. The specifications differ mainly in the level of their requirements. In some cases their different audiences also require a different structure and an emphasis on different aspects of the requirements.

3.3 Audiences and contributors

The audiences of an RFT specification will include the following groups of people. This list is not complete - there will be other audience groups who have an interest and an investment in the specification.

a. **Contractor personnel**. The primary "user" of the specification is the tendering contractor. Relevant personnel will include those who decide whether to tender, those who prepare the tender, and development personnel who will advise the others. Most are stakeholders in the project.

b. **Operational users**. The potential operational end users of the system need to understand the specification so that they can assure themselves that it represents their needs. The fact that the specification is derived from and traceable to the higher level requirements defined by user representatives helps in this regard but is not sufficient. Development of the specification will have involved the rewording of requirements and derivation of more concrete and detailed requirements which need to be validated.

c. **Support users.** The word "users" has been extended in the systems engineering field to include those who support the system (see section 2). These users will have different requirements from those of the operational users; otherwise the same comments apply.

d. **Project management personnel**. Because the specification is the single most important document in shaping the system under procurement, the Project management personnel need to understand both the requirements and their likely consequences.

e. **Engineering personnel**. The technical specification is often produced by engineering personnel (in Naval Engineering Services). Many engineers will be involved in the definition of requirements, although the actual specification may be coordinated and drafted by less than five of them. All contributors need to ensure that their area of expertise is correctly represented, in the context of the total requirements.

f. **Trials and test personnel.** The personnel who will have responsibility for the operational evaluation of the system are also a potential audience. Although acceptance into Naval service will be based on the Detailed Operational Requirements (see section 3.2) and contractual acceptance will be related to the Contract specification, such personnel can offer important advice with regard to the verifiability of requirements.

g. **Specialists**. There will also be specialists outside Naval Engineering Services and outside Navy who will contribute to the requirements. Specialists may address such diverse areas as software upkeep, security, safety, logistics, interoperability and the specialised technical areas. External contributors may include DSTO and contractors supporting the Project, for example.

All of these audiences will need to understand the specification, or at least their special areas of interest, both in ensuring that the specification is adequate, and in carrying out the evaluation.

Our experience has shown us that it is impossible to write a specification that will completely satisfy all its audiences. As shown in the industry survey, many contractor personnel prefer a mechanical approach, so that the specification may be easily parsed, and requirements traced. Audiences who are less familiar with specifications prefer a descriptive style, where each area is complete in itself. They dislike cross references to other parts of the specification. Specialists may prefer a specification style which is commonly used in their area of expertise.

Broadly, the audience can be divided into two groups: the tenderer, who is the primary "user" of the specification, and the Navy personnel (and others) who contribute to the content and quality of the specification, either by contributing to it directly, or in helping to validate it - the "contributors". This latter group is extremely diverse, as shown above. It is also worthwhile noting that the tenderers may also be seen as contributors if they are provided with pre-release drafts of the specification.

The needs of the tenderer were examined in the industry survey. Many of the problems identified applied to the clarity, correctness, level and structure of requirements.

The needs of the "contributors" can be summarised as follows:

- The specification must be understandable and readable to a wide variety of personnel.
- The structure of the specification should appear logical, provide an overview of the specification at a glance and allow the relevant requirements to be found quickly.
- Requirements addressing common areas of interest should be collected together.

As discussed in section 1.3, correctness and completeness of the specification are the prime objectives. To reach this objective, the needs of the contributors must be the main consideration. Only they can accurately identify and validate the requirements.

Writing these requirements in a form which also adequately communicates the requirements to the tenderer can be regarded as a separate problem. This requires a small number of requirements engineers who are trained and skilled in this task.

3.4 Alternative specification methods

3.4.1 Introduction

There are hundreds of books and articles which provide some advice on developing specifications. Many of these advocate practices which are far

removed from the natural language specifications used by Navy. This section examines some of these practices as contenders for RFT specifications.

Throughout the world, natural language is still the most common specification method for all levels of specification, and particularly for high level specifications [Mar 1994; Davis 1993a]. The use of natural language certainly leads to problems, because of the impossibility of stating needs in a totally unambiguous way. This has led to the search for different methods of specification which remove or reduce the ambiguity.

It should also be noted that, although natural language specifications have weaknesses, they have been used in thousands of successful procurements over a long period of time.

3.4.2 Development specifications versus RFT specifications

Before addressing some of these methods, it is important to introduce a note of caution. The majority of texts on specifications address "development specifications" rather than RFT specifications, without stating this directly. Development specifications are usually written by the system suppliers to define the requirements for a system or a system component which they will develop. Its needs differ significantly from those of an RFT specification. In many cases the supplier already has some views on equipment selection and high level design, and the specification will reflect these to some extent. The requirements will also be written with an understanding of the supplier's resources and development methodologies. Some of the requirements will have been derived so far from the original user requirements that many users would not understand them.

An example is a specification for a parallel computer bus. Most of its requirements are at the level of bandwidth, peak and average data rates, and contention minimisation. The specification may refer to these as "user requirements". While this *is* a requirements specification, it is obvious that many decisions have been made since the users defined their requirements, and that this specification is no longer in the user domain. This is a development specification.

3.4.3 Alternative methods

As stated above there are numerous documented analysis and specification methods. The methods presented here are representative of most of the better known ones. It should be noted that none of the methods below is ideal for all application areas. There is strong support for all of these methods in the literature, and also strong criticism of their scope of useful application. As Davis [1993a] states, the claims for some of these methods are insupportable.

a. **Logically formal methods.** Although the phrase "formal methods" is used widely in the literature, there is little agreement on what "formal" means. We will use "logically formal" for methods which have a basis in

mathematical or logical proof, and "semi-formal" for other methods which use a symbolic and disciplined approach.

Logically formal methods are most often used when it is necessary to prove that low level functional requirements are complete and unambiguous, and that software will perform a predefined task with no side effects. These are normally used for systems which require a high level of trust, such as safety critical systems. Typical examples use mathematically based languages such as Z, and rule based languages such as Prolog. The promise of proving completeness is attractive, but completeness can only be proven in a quite restricted zone of performance. The problem of proving correctness and completeness of higher level requirements is not currently solvable by logically formal methods.

In addition to this limitation, logically formal methods are yet to gain wide acceptance in the systems engineering community, and are generally practiced only on small scale applications [Zucconi 1992], or relatively small elements of a larger system.

b. **Semi-formal methods.** Semi-formal methods use a disciplined representation of requirements which provides a more expressive (and readable) language than logically formal methods, but which avoids much of the ambiguity of natural language specifications. Such methods include finite state machines, decision tables and trees, specification description languages and program design languages. There are also various "packaged" methods which offer an integrated approach to analysis and design, e.g. Jackson System Development and Ward-Mellor methods. Many of the graphical methods referred to below can be also be regarded as semi-formal methods.

c. **Graphical methods**. Graphical methods represent the requirements in graphical form, using a systematic notation. Many of these methods are proven and widely used, particularly in information system analysis and design. Examples include Data Flow Diagrams, Entity Relationship Diagrams, Statecharts, Petri Nets and N^2 Charts. All of these methods are associated with some other form of descriptive specification, typically natural language. While such methods are commonly used in the analysis of requirements and in design, their use in user requirements specifications, particularly for combat systems, is rare.

d. **The user manual as a specification.** One method of specifying a system is to write a user manual for the eventual system, and to use this as the basis for design [Howes 1990]. While this has been shown to be a useful approach for the development specification for some systems, it is clearly unsatisfactory for higher level user specifications, where it is important to concentrate on the users' needs rather than the design of the solution.

e. **Using a prototype as the specification.** In this approach the users participate in the design of a series of prototypes, which eventually represent the user interface, functionality and performance they are

seeking. There are a number of problems in this approach for RFT specifications:

- The method concentrates on design rather than requirements definition.
- The prototype or prototyping environment will have limitations which may influence the design.
- Although the prototype may appear to be a good model for the required system, it would be unreasonable to expect every aspect of its functions and performance to be exactly replicated in the final system. Identifying such areas of performance variations can be very difficult.

Prototyping as an aid to requirements capture and analysis is discussed further in section 4.3.6. There are many aspects of a complex system which cannot be adequately defined by a prototype, including complicated processing algorithms, non-functional requirements and support requirements. These would need to be specified by a different method.

f. **Executable specifications.** Unlike prototypes, executable specifications provide a model of the requirements written in a formal language which can be executed to investigate the quality of the requirements [Fuchs 1992]. Although this approach shows promise, it is currently immature and is unlikely to be able to model all the requirements.

g. **Object oriented analysis (OOA).** Although not a specification method, object oriented analysis results in a specification which is quite different from that resulting from the more widely used functional decomposition methods. This approach is based on identifying objects in the application area (some of which are abstract) and defining requirements for the actions performed on those objects and relationships between them. The general effectiveness of using OOA is as yet unproven and hence OOA, although currently used effectively in some types of applications, must be regarded as an immature method.

Each of the above methods (apart from object oriented analysis) attempts to reduce the problems associated with natural language specification. In general their strengths can be seen to be:

- A less ambiguous statement of the requirements.
- A more consistent specification.
- A simpler approach in some cases (where the requirements lend themselves to such treatment, as in the detailed requirements for a text editor, for example).

On the other hand, there are several disadvantages in using these alternative methods for the specification of high level user requirements.

- Without special training, the specification is generally much more difficult to read and understand than a natural language specification [Mar 1994; Davis 1993a]. This particularly applies to logically formal and semi-formal methods.

- There is an inherent tendency to trust something which is presented in a highly systematic manner, particularly if it is difficult to understand, by analysts who claim to understand it. This will adversely affect validation of the requirements by users and other contributors to the specification.
- There is a risk of anticipating the design too strongly, or influencing the design towards the structure of the requirements model.
- They are generally not suitable for the specification of non-functional requirements [Zave 1990; Mar 1994; Roman 1990].
- They tend to be weak for the specification of performance (as opposed to functional) requirements [Zave 1990].
- They are generally restrictive in their ability to specify the functionality of a system, which may result in concentrating more on requirements which match the method and neglecting those that do not.

One proposed solution to the lack of readability of such specifications is to translate the specification into natural language for those not skilled in the method, either automatically or manually. This ignores the fact that the main reason for using such methods is the ambiguity inherent in natural language specifications, and which will clearly apply to the translated version, and also ignores the risks in using more than one specification for the same set of requirements. Such specifications are also likely to be extremely mechanical in both language and style, reducing their readability [Davis 1993a].

While this section has been pessimistic about the value of alternative specification methods for specifying user requirements, it is evident that most of the methods provide strong benefits in specific circumstances. It is likely that these circumstances will apply to some Navy applications, or parts thereof. An example is where the protocol for operational procedures can be comprehensively defined by the user, such as in message processing. In any case, it is unlikely that any single method will be appropriate for the entire application, and a mixture of methods will probably provide the best result [Zucconi 1992; Mar 1994; Hofmann 1993].

It is obvious that an extraordinarily large amount of effort has been spent on the analysis and development of alternative specification methods. It is interesting to speculate on how natural language specifications may have benefited if a similar amount of effort had been applied to their improvement. Similarly, it would be interesting to compare the quality of a logically formal specification (say) prepared by a novice in the method used, to the quality of a natural language specification prepared by a skilled requirements engineer. We are unaware of any research in this area.

3.5 Specifications and risk

The level of specification should be decided on the basis of risk, where risk reflects both the adverse consequences (the "impact") of an event occurring and the probability of its occurrence. Where there is little risk of developers not providing an adequate solution, the requirements can be defined at a relatively high level. It is evident therefore that the risk will depend on the requirement itself, how it is expressed, and the developer's ability and experience. The systems examined in this study, moreover, are complex and prone to volatility

in their requirements, both of which are contributors to risk [Kirkpatrick et al. 1992].

Risk management is conceptually simple. It involves identifying all of the potential risks, agreeing methods for their mitigation and control, and then implementing those methods while the risk still exists. In practice, of course, risk analysis is very difficult, mainly because of the skills and experience needed to identify the risks in the first place. Reaching agreement on what constitutes a risk can also be a stumbling block.

One of the objectives of this study is to help identify the adverse impacts of poor specification practices, so that risks are more likely to be identified. To be truly useful it is important that the development of specifications is regarded as a risk oriented activity. Risks need to be identified very early in the definition process, and the process adapted to manage those risks. In many cases, the specification development team will have little experience in risk analysis and a limited ability to identify the future problems. In these situations the risk analysis should be performed by risk specialists (typically requirements engineers) in conjunction with the specification team, the users and the project staff.

The following are seen as typical sources of risk:

a. **Uncertainties in the operational requirement.** This situation occurs more often than might be expected, particular when the requirement is new or users have little experience in the relevant technologies, and can be a critical source of risk. In these cases it is impossible to specify requirements at a detailed level with any confidence, and it should not be attempted. Alternative strategies for managing risk might include:

   - Defining only the operational requirements, but in as much detail as possible, and relying on the developer's experience to resolve the problem.
   - Undertaking specific exercises to provide a better understanding of the operational requirement. This might include the use of models or prototypes, or seeking the assistance of others who have such experience, possibly from other countries. These are usually conducted as project definition studies.
   - Including specific contract activities to reduce the risk, such as including user representatives in the developer's design activities.

b. **Drafter inexperience.** This risk can only be managed by frequent review by experienced drafters. It is not sufficient to review the specification only when it is substantially complete. Not only will the specification be more difficult to change at a later stage, with critical problems of structure for example, but frequent reviews will provide a beneficial learning experience for the drafters.

c. **Developer inexperience.** As was stated above, the specification risk will depend in part on the experience of the contractor chosen to develop the system. As the industry survey showed, developers with limited experience in a particular aspect of the system preferred a lower level of

specification in that area, to reduce their own risk. Unfortunately, the developer is rarely known at the time the specification is written, and in any case assessment of a developer's experience is prone to error. Where the selected contractor has less experience than expected, this risk must be addressed prior to the contract being let.

3.6 Performance based specifications

There has been strong emphasis placed in recent years on the use of performance based specifications for the definition of requirements for the Request for Tender (RFT), by both Navy and Defence. Performance based specifications have also been suggested for contractual specifications.

Performance based specifications concentrate on the required functions and performance needed, with other non-functional requirements being carefully scrutinised to ensure that they do not unnecessarily restrict the field of possible solutions and solution providers. It was clear to us in our surveys that even these definitions do not guarantee a common understanding of what constitutes a performance based specification, with some proponents seeming to equate "performance based" with "short". Quoted examples such as a 3 page specification written in 1907 for an aircraft [Kalms 1990] do not help this situation, considering the 1000 fold increase in complexity in equivalent products today.

Most Navy RFT specifications for complex systems in the recent past have been performance oriented, with some notable exceptions in specific areas. There is evidence that the practice in this area has improved significantly. Our industry survey and examination of current projects (Reports 1 and 2 in this series) showed examples of what were perceived by both industry and projects as overspecification and underspecification, although there was rarely unanimous agreement. There were also suggestions that the current policy towards performance based specifications had increased the risk to Navy in some cases.

Many of the comments in this section are based on the following references:

- The *Guide for the Preparation and Use of Performance Based Specifications* [Millett 1994], prepared for the US Army Materiel Command, provides a strong argument for the use of performance based specifications. It provides little firm advice on how to write them, however.
- *Commercial Practices for Defense Acquisition* [Rhoads 1992] is a Defense Systems Management College (US) Guidebook which examines the differences between commercial and US defence procurement methods.
- *Lessons learnt on the JP2030 Project* [Davis 1993b] is a DSTO report which examines the requirements phase of a project to develop a Joint Command Support Environment for the Australian Defence Force.
- *Software requirements: objects, functions and states* [Davis 1993a] is a comprehensive and practical guide to requirements engineering, particularly for software systems.
- Lessons learnt reports from Navy projects.

Performance based specifications offer a number of advantages.

- They allow the developer to address the real problems rather than the customer's perception of how the problems should be solved.
- They can result in lower project costs due to the contractor's flexibility in choosing between a number of solutions.
- They encourage innovative solutions which may not have been anticipated by the customer.
- They encourage the use of existing equipments, software and components including commercial off the shelf (COTS) system components.
- They may increase the number of suppliers who can effectively tender.
- It is less necessary to resolve whether a derived (lower level) requirement is mandatory or not - these issues tend to be resolved in evaluation, contract negotiation or preliminary design.

We encountered some strong reluctance to the use of strictly performance based specifications in our survey of projects. Millett [1994] identifies some of the reasons for this reluctance:

- The customer often knows better than the developer what detailed functions are required.
- Increase in risk.
- The ability to encourage standardisation is reduced.
- Difficulties in evaluating high level specifications.
- Anxiety in forming a close relationship with the developer, which may be necessary in such cases.
- It is preferable to use proven project management procedures rather than adopt radical approaches.
- It is easier to cut and paste from previous projects than concentrate on getting the correct performance based requirements.
- Ignorance of the advantages of such specifications.
- Lack of guidelines in the preparation and use of performance based specifications.

While the perceptions listed above may or may not be correct or relevant in all cases, there are potential risks and penalties in using performance based specifications, which must be considered when deciding the level of specification needed.

- Systems developed by inexperienced or unprincipled contractors may result in worse products than those developed to a detailed specification.
- Performance based specifications often reduce the developers' risks. It is not surprising, therefore, that some developers are advocating such practices.
- Performance based specifications increase the risk of an *inadequate* system.
- Performance specifications are difficult to write such that they guarantee the required performance. Most personnel do not have the skills to derive performance based requirements from operational requirements to the necessary level of definition. In addition, some contractors will attempt to constrain acceptance testing only to the high level requirements, so that the testing of important lower level functions (derived by the contractor) is less visible to the customer and not subject to acceptance controls.

- Performance based contracts may require more, and more skilled, project support effort from Navy than for detailed contracts, rather than less as is sometimes assumed.
- High level specifications increase the risk of dependence by Navy on implicit requirements.
- Performance based specifications may reduce the opportunity for Australian companies to participate in Navy contracts, due to their relative inexperience in comparison with international suppliers. Local suppliers are less likely to be able to offer comprehensive and sophisticated solutions in specialised areas without strong guidance from Navy.

The level of specification needed will depend on whether Navy is selecting a supplier in the RFT, or a product, or both.

The problems in adopting an approach to this decision is best shown by an analogy that many of us face, that of building a house for private use. It would be possible to develop a "performance based specification" for the house (although few of us do) showing the size of our family, what we intend to use the house for, the environment in which the house is to be built, and so forth. We would normally add some "non-functional" requirements as well, addressing issues such as preferences for building standards, the expected life of the house, maintainability issues and some personal aesthetic preferences. We would probably put no constraints on the strength of the walls or how the rafters are to be arranged in the roof. We would expect building regulations and standards to reduce our risk in this area.

We would then approach architects and/or builders, asking for their proposals in meeting our requirements, including cost and schedule. We may even pay additional fees to short listed tenderers to have detailed plans drawn up. After evaluating the proposals, discussion with the tenderers and contract negotiation, we would sign a contract to build the house, on the basis of the proposed design.

There is little doubt that this is a sound and proven way of doing commercial business, particularly when we are protected by numerous industry standards, regulations and builder licensing schemes. We also have reduced the risk of an inadequate house by choosing both the supplier and the product, or rather a preliminary design including a combination of some requirements and products.

A strictly performance based contract, on the other hand, would only include our original requirements, modified to some extent by our experiences in the meantime. The tendered design would effectively be discarded. The builder would be free to redesign the house within the constraints of the specification, to minimise his costs. It is unlikely that many house buyers would find such an approach acceptable, although some builders would be strongly in favour of it.

It is not surprising that this is not common practice in commercial procurement either. Our industry survey showed that most subcontracting in Defence projects was on the basis of either a product or a design rather than high level

requirements. Commercial practice for development was addressed by Rhoads [1992]:

> They rarely contract out for new development, relying instead on internal research and development, or joint development with a supplier. Because of this, commercial companies have little need for detailed process specifications.

One advantage commercial companies have over Navy customers is that when entering into a joint development, both parties have a similar commercial motivation, making tradeoffs between them much simpler.

We strongly support the use of performance based RFT specifications in Navy projects for complex systems, but believe that there is currently insufficient understanding of what "performance based" means, or of the potential risks in using such specifications incorrectly.

3.7 What specification method should Navy use?

On the basis of the above we conclude that the natural language performance based RFT specifications currently used by Navy are appropriate for Navy's needs, and in most cases will be more effective than any of the alternatives.

It is possible that some of the proven semi-formal methods in particular will be preferable for parts of some specifications, but only where it is necessary to provide detailed specification of known processes. These situations are relatively simple to identify - they occur where natural language specification would appear stilted and repetitive, or where there are serious difficulties in expressing requirements clearly using natural language. Examples include message handling and data link protocols. In these cases a proven and widely used representation, such as state tables or data flow diagrams, should be used where possible. In some cases however, it may be preferable to use a notation which is commonly used by the potential users of the system.

Otherwise, the number and wide diversity of the audiences of RFT specifications argue against the use of other than natural language. It would be impractical to train all of these personnel adequately in the methods used, resulting in a reduced readership and hence a lower level of validation of the specifications. Where training is to be provided, it is probably better utilised giving personnel an improved understanding of requirements engineering and the problems which may be incurred in the implementation of the system. It should also be noted that relatively few personnel will have a continuing responsibility for requirements.

With regard to many of the other perceived problems identified in our surveys of contractors and projects, we consider that the majority of these can be rectified by concentrating on the development process for requirements and specifications. These problems and recommendations for their solutions are addressed in the following sections.

4. Requirements

4.1 What Are requirements?

A requirement is a capability needed by a user to solve a problem or achieve an objective. When expressed in words or other notations, usually in a specification, it is also the statement or definition of that capability. These two definitions are referred to as the "genuine requirement" and the "written requirement" respectively. Specification drafting involves translating the genuine requirement, the needs of the users, into a set of requirements called a specification. It is important to realise that this translation will always be imperfect; ideas and concepts are usually impossible to represent completely and accurately in words, or in any other notation.

During the course of a project, there will typically be a number of specifications generated, with additional requirements derived from, and traceable to, the original user requirements.

Section 2 contains definitions of and discussion about the different types of requirements, additional to the material in this section.

There are different types of requirements, each with their own characteristics:

a. **Primary and derived requirements.** A primary requirement is the original statement of an individual requirement in the specification hierarchy. All other requirements are derived from the primary requirements or from other derived requirements.

   This does not automatically mean that all requirements in the RFT specification are derived, directly or indirectly, from requirements in the highest level document (although most will be). In some cases, either due to omissions in higher level specifications, or due to a difference in the purpose of the different specifications, primary requirements will be introduced in lower level specifications. One example could be a requirement for compatibility with equipment which already exists in Navy ships. This requirement may not based on operational needs but instead on reducing the overall maintenance and repair costs within Navy.

b. **Functional, performance and non-functional requirements.** A functional requirement is a description of some function that the system is required to perform. Performance requirements define how well the system is to perform a specific function. Non-functional requirements describe the constraints under which the system must operate including factors such as availability, reliability, and maintainability.

c. **Mandatory and non-mandatory requirements.** Mandatory requirements are requirements that the tendered system must meet, and are specified using "shall". Non-mandatory requirements are lower priority requirements which are not considered essential, and are usually specified

using "should" or "it is desirable that". Often non-mandatory requirements are used to indicate a preference that may be used as guidance by the tenderer in the selection of equipment or components. Evaluation of tenders is based primarily on the mandatory requirements, with the non-mandatory requirements used to discriminate between tenders which are close in capability.

d. **Explicit and implicit requirements.** The written requirements may all be regarded as explicit requirements. Implicit requirements, on the other hand, are requirements which may be inferred from the specification but are not actually defined. Because of differences in experience between different readers and the specification drafters, what is inferred by readers will not always be what was implied by drafters, leading to tenders which may be strictly compliant but which do not meet the genuine requirements.

   It is impossible to eliminate implicit requirements. As a gross example, we have never seen a requirement that a ship should float (that requirement is implicit in the word "ship") although it is a genuine requirement. Implicit requirements can be a serious source of risk, however, and need to be carefully monitored.

4.2 Requirements engineering

Requirements engineering is a fundamental part of systems engineering concerned with defining and managing requirements, particularly for complex systems. It emphasises the importance of correct and consistent requirements, and the need for planning and analysis prior to the drafting of a specification, to achieve this end.

Any product or system is critically dependent on the specification from which it is derived. Unfortunately requirements for complex systems are difficult to capture and specify for a number of reasons:

- The detailed requirements for a complex system are numerous and interrelated. It is therefore difficult for anyone to confidently predict all the requirements.
- During a long project life cycle, requirements will inevitably change.
- It is difficult to get users' enthusiasm and support at the requirements stage when implementation is a long way off.
- The capabilities of technology are increasing rapidly and are difficult to predict.

Requirements engineering embraces a number of activities, which are interrelated and tend to be carried out iteratively rather than each activity being performed in isolation [Millett 1994]. The activities can be grouped as follows:

- **Requirements capture** is the activity of eliciting the requirements which encapsulate the users' needs, usually from the users. This is also called requirements elicitation.

- **Requirements analysis** is the study of the captured users' needs to derive a coherent, complete, consistent and feasible set of requirements. This often includes modelling and tradeoff studies.
- **Requirements definition** involves representing the requirements in a usable form, usually by drafting a specification.
- **Requirements validation** is the process by which the requirements engineer ensures that the written requirements are an adequate expression of the genuine requirements.
- **Requirements tracing** is an activity which attempts to ensure that:
    - Every new requirement (at any level) is justified and validated, and
    - Every other requirement and product feature in the specification tree is derived from and traceable to a higher level requirement, and vice versa.

  This should ensure that any requirement can be tracked to the product feature/s which satisfy that requirement. It should also ensure that each product feature can be justified by the requirement/s which caused it to be incorporated in the system. Requirements traceability is essential for efficient verification.

Requirements engineering is applicable to all stages of a project, with the capture, analysis, definition and validation activities being most intense in the early stages. These activities are also necessary in the latter stages, however, as requirements are refined or changed.

Although there are several packaged methodologies that address part or all of the requirements engineering process for developing RFT specifications, they are not widely used for this purpose and most tend to be used later in the development cycle. As Gause and Weinberg [1989] point out, strict methodologies and tools do not solve the problem, although they can make the presentation of the results easier, and can sometimes provide viewpoints that might otherwise be unexplored. Examples of such methodologies are CORE, SSADM, JSD and OORA.

4.3 Capturing user requirements

Capturing user requirements is generally recognised as both a critical activity and one which can be extremely difficult [Fairs 1992; Frantz 1993; Black 1989]. Gause & Weinberg [1989] provides a comprehensive and highly readable reference for those interested in improving this important area.

Potential system users are obviously the most important participants in determining the user requirements. Unfortunately, they may be unsure of what they need, and they often have difficulty expressing those needs in a suitable form for system design and development. Even when they believe that they clearly know their needs, those needs may prove to be conflicting, vague and incomplete. The importance of obtaining the correct requirements requires techniques where users are an integral part of a requirements capture team, that includes skilled requirements engineers. Strong contribution by users will also increase their sense of ownership in the requirements which is important in

ensuring their responsibility in later contractual and development activities [Black 1989; Fairs 1992].

It is important that *all* types of users contribute to the determination of requirements, not only the operators of the proposed system. In particular, maintainers and those who will provide shore based support should also be included (see section 2).

The most effective capture strategies are discussions with potential users, watching them undertake the tasks in question using their existing facilities, and allowing users to experiment with early versions (or prototypes) of the system [Davis 1993a]. Techniques that implement these strategies include structured interviewing, storyboarding, brainstorming and prototyping. Each has its own areas of application and associated risks and they are discussed in some detail below.

There are also several requirements engineering techniques that are the subject of ongoing research and although not widely used at present do hold some promise for the future. Knowledge engineering and formal methods are examples of long term research topics.

4.3.1 The user-engineer relationship

While there is generally a close and cooperative relationship between engineers and operational user representatives in Navy projects, with both sharing a common objective, the backgrounds of the two groups results in them having different priorities and viewpoints. This has the potential to cause conflicts during the requirements phase, often due to an "us" versus "them" situation developing. Understanding the different viewpoints can reduce such conflicts. Table 2, based on Scharer [1990] summarises potential difficulties the user-engineer relationship.

Table 2. *The user-engineer relationship*

How engineers see users	How users see engineers
Users:	Engineers:
Don't really know what they want	Don't understand the operational needs
Can't articulate what they want	Place too much emphasis on technicalities
Have too many needs which are politically motivated	Try to tell us how to do our jobs
Want everything right now	Can't translate clearly stated needs into a successful system
Can't prioritise needs	Say no all the time
Refuse responsibility for the system	Are always over budget
Are unable to provide a usable statement of needs	Are always late
Are not committed to system development projects	Ask users for time and effort even to the detriment of the users' important primary duties
Are unwilling to compromise	Set unrealistic standards for requirements definition
Can't remain on schedule	Are unable to respond quickly to legitimately changing needs

Such attitudes can lead to an environment in which it is impossible to adequately capture the needs of the users, or where the users will feel they have

a reduced ownership in the requirements. It is essential that all parties participating in the requirements capture understand and appreciate the contribution that the others are making to the project.

4.3.2 Choosing the right users

There is no simple formula for choosing the right users for determining the user needs. A sufficient number of users should be involved, however, to cover the main functional and support areas of the system and they should come from a range of levels within the organisation. The basis for choosing individual users should be as follows:

- **Users have to be knowledgable.** They need to have extensive experience in their particular area of expertise.
- **Users have to be articulate.** They need to be able to express their requirements effectively although to some extent they will be assisted in this by experienced requirements engineers. They need to have reasonable communication skills.
- **Users have to be enthusiastic.** They have to want the new system to succeed and concur with the course the project is taking.
- **Users have to have time.** Involvement in a complex project is a serious commitment and can consume a large amount of time, sometimes at short notice. Users' contributions to the project needs to be planned and assured, and reconciled with their other work commitments to ensure their availability as required.

4.3.3 Structured interviews

Usually conducted by professional requirements engineers, structured interviews involve discussing the requirements with potential users stimulated by a set of carefully considered questions. Questions should be chosen that do not suggest answers to the users, and which stimulate users into thinking about different viewpoints of their requirement. Although the formulation of questions and the skill of the interviewers is important, success of the interviews will largely depend on the choice of users as discussed above.

4.3.4 Brainstorming

In this approach a number of users assemble to discuss requirements and an analyst acts as facilitator. The interaction between the various users acts as a catalyst to enhance the flow of information and ideas. The role of the facilitator is to stimulate the discussion, to prevent participants getting sidetracked, and to ensure that discussion is focused on the requirements rather than possible solutions.

One possible inhibitor to brainstorming is that it intentionally encourages freedom of thought and expression, mainly by not restricting ideas or by analysing them in detail. Its strength is in gaining a large number of ideas, many of which may not be feasible or which may be downright silly, but which may stimulate other thought processes towards useful concepts. Some

participants may find this environment confusing, non-productive or even threatening, and will therefore be less than useful.

Brainstorming does not, in itself, capture the requirements. After a brainstorming session it is necessary to filter and analyse the ideas that have been generated and use them as the basis for more structured discussions, such as in structured interviews.

4.3.5 Storyboarding

Requirements capture is an iterative process. Storyboarding can be used to check that requirements are correct and suggest options to users. With this technique user requirements are illustrated in a series of displays that are then linked to simulate both the requirements and how the system will behave once developed. The entire storyboard (which might consist of 100-300 displays) is then demonstrated to users who are asked to comment [Andriole 1990]. Users can have some measure of interaction with the storyboard simulating operation of the system, albeit less effectively and more slowly. The users' comments are noted and used to update the storyboard and eventually the requirements document.

One disadvantage of storyboarding is that it is easy to promise users unrealistic solutions since the storyboards are so easy to create. Storyboarding is a special case of prototyping and also has some of the same risks.

4.3.6 Prototyping

Prototyping is a powerful technique that can be used to refine requirements. A prototype in this context is a working model of a system or subsystem, which emphasises specific parts of that system. There are many different types of prototypes with applicability to different phases of the system development life cycle. Vonk [1990], Lichter et al. [1994] and Davis [1993a] provide a good introduction to the different techniques and uses of prototyping.

Generally prototypes can be divided in two categories: throw away and evolutionary.

- **Throw away** prototypes are "rough and ready" models of specific system functions built quickly to demonstrate functions or solutions to users.
- **Evolutionary prototypes** are in fact early implementations of the system, concentrating on specific needed functions, which will eventually evolve into a final system. They therefore need to be developed to the standards required of a production system. Pilot systems are an example of evolutionary prototypes.

Prototypes are particularly useful when the users are not conversant with the functionality which might be available [Davis 1993b]. They also allow the users to see and experiment with potential solutions to their requirements, usually resulting in changes to those requirements, and the generation of new requirements. These changes can then be incorporated in the prototype - this is usually called iteration - so that they can be validated. Prototypes have the

added advantage of increasing user participation and enthusiasm by providing an early taste of system functions.

Typical uses of prototypes in the requirements phase are to:

- Determine the feasibility of a requirement.
- Validate that a particular function is really necessary.
- Uncover missing requirements.
- Determine the viability of a user interface.

Undisciplined use of prototyping can be detrimental to requirements capture and analysis, mainly because it tends to be solution oriented rather than problem oriented:

- Because the prototyping offers a solution, there is a temptation by both users and requirements engineers to concentrate more on designing the system than in capturing and analysing their requirements.
- When faced with a prototype many people accept the implementation as being the only way to meet their requirements. A prototype may therefore restrict the development of requirements to the basic architecture of the prototype.
- The prototype or prototyping environment may have limitations which influence decisions regarding requirements and potential solutions.
- There are many aspects of a complex system which cannot be adequately defined by a prototype, including complicated processing algorithms, non-functional requirements and support requirements. It is important that focusing on the prototype does not allow these important areas to be neglected.

Perhaps the most critical pitfall of prototyping is the common belief that using a prototype replaces the need to gather requirements by other means. This is not true. A good understanding of the requirements is necessary to design the prototype in the first place. If this information is not available there is a high probability that initial guesses by the prototype designers will strongly influence the requirements being generated using the prototype [Lichter et al. 1994].

Despite these risks, we strongly endorse the use of prototypes for refinement and validation of requirements, *after* the requirements have been captured, analysed and defined by other means.

4.4 Requirements validation

Validation is the process of determining that project products meet the users' needs (see section 2) and can be critical to project success [Black 1989]. During the requirements phase each specification has to be validated against both the higher level specifications from which it was derived (such as the Detailed Operational Requirements), and against the unwritten needs of the users.

Ensuring the traceability of the requirements to higher level specifications is the first step in the validation process. This shows that the requirements conform

to those defined at a higher level. Testing the requirements against the genuine needs of the users is more difficult.

There are various methods for undertaking validation ranging from manual review, usually by reading the requirements, through to the use of prototyping. The circulation of specifications for review by stakeholders is a passive form of validation which is generally regarded as being of limited effectiveness [Black 1989; Davis 1993b]. Proactive methods such as prototyping are more likely to achieve the necessary assurance [Hofmann 1993; Black 1989]. The effort and time involved in setting up a formal review process will be worthwhile, considering the potential consequences of inadequate specifications (section 1.3). Table 3 from Boehm [1990] suggests different validation methods.

Table 3. *Different requirements validation techniques*

Simple manual techniques	Reading Manual cross-referencing Interviews Checklists Manual models Simple scenarios
Simple automated techniques	Automated cross-referencing Simple automated models
Detailed manual techniques	Detailed scenarios Mathematical proofs
Detailed automated techniques	Detailed automated models Prototypes

Validation audits need to be planned and risk based, with additional effort being devoted to those areas of the requirements which are known to be either contentious or difficult to define. Best results will be achieved from the validation if it is carried out by an independent team. A typical process for specification validation is as follows [Boehm 1990]:

- The validation agent analyses the specification and issues problem reports to the specification drafter.
- The drafter isolates the source of the problem and proposes a correction to the specification.
- The project and user representative approve any proposed corrections that would perceptibly change the requirements baseline.
- The validation agent analyses the correction and issues further problem reports if necessary.

Systematic and visible validation of the requirements should also provide fringe benefits, including a greater confidence that the requirements are right, reducing the influence of dissenting parties in Navy and Defence. It will also give materiel personnel a better understanding of why requirements exist, providing a sound basis for system evaluation and tradeoffs. McNaugher [1989] shows that in some cases there are pressures to include requirements which are not representative of the users' needs but which are more intended to drive the system selection and design in a desired direction. Serious validation should detect such false requirements. McNaugher states:

> [Specification contributors] are trying to protect the specific capabilities or technologies that matter to them. The more they can embed their own project goals in detailed specifications, the surer they are that those goals cannot be traded away during development.

4.5 Requirements management

Requirements management is the configuration management of requirements throughout the project. An RFT specification typically contains hundreds of individual requirements. When a change is made it is essential that the total context of the requirement is considered, to ensure that traceability is maintained and that other related requirements, in all relevant specifications, are identified. In essence, requirements management involves:

- Ensuring that there is a single authoritative definition of the requirements, including annotations, justifications and traceability links to other specifications.
- Controlling changes to the requirements, including validation and authorisation of changes.

It is preferable that a systematic process is established for the management of requirements, aided by the use of computer based tools (see sections 8.2.2 and 8.2.3).

Modifying requirements requires as much skill and care as defining the requirements in the first place. We have seen several examples where "small" changes have been made by persons not involved in the original specification development, sometimes for cosmetic reasons, with potentially serious adverse results. Specifications are similar to computer programs in that small changes can have disastrous effects on the effectiveness of the specification as a whole, sometimes in areas supposedly remote from the area in which the change occurs.

It is also important to realise that changes *will* occur, even after the specification is regarded as complete, and to cater for or even anticipate those changes. While it is important to have a stable agreed requirements baseline, a strict policy of "freezing" requirements at a certain stage can often have effects worse than those of a moving baseline [Freedman & Weinberg 1990; Humphrey 1990], particularly where the requirements are uncertain or immature.

4.6 Requirements and evaluation

The RFT specification is the basis for the evaluation of the tendered technical proposals. As such, the representation of requirements can be important in the effectiveness and efficiency of evaluations. Our project survey showed that the types of specification used are adequate for evaluation purposes.

The following comments are made with regard to specification issues which affect evaluation.

a. **Verifiability.** Requirements must either be verifiable against a proposed solution, or should facilitate direct comparison between proposals. Some requirements are inherently subjective.

   Consider "The wordprocessor shall be a widely used COTS product with a modern user interface". While it might be difficult to derive objective tests for this requirement, there is no doubt that the requirement is genuine. Deriving additional requirements such as "cut and paste", "window based" and "fonts including ..." may adversely restrict the choice of wordprocessor. It is likely, moreover, that objective users could reach close agreement on the ranking of potential candidates and would agree, for example, that Microsoft Word 6.0 is preferable to Wordstar 3.0 against this requirement.

   Other requirements such as "The system shall facilitate the use of normal Navy practices and procedures, with minimal modification" are far less verifiable, and provide insufficient guidance to both the tenderers and the evaluation team.

b. **High level requirements**. High level requirements, which are also qualified by detailed requirements, are difficult to evaluate, and provide an additional problem with regard to precedence (see section 6.6). It may be preferable to state these as system objectives (using "will" rather than "shall"), rather than as requirements.

c. **Low level requirements**. Some low level requirements, while valid, have no real effect on the evaluation result, and hence need not be individually evaluated. Such requirements are not difficult for the contractor to provide in any case, and can usually be resolved adequately during contract negotiation. It is not suggested that these requirements be removed from the requirements specification, merely that they not be evaluated.

d. **Implicit requirements**. Implicit requirements can be a serious source of risk (see section 4.1). Considering the needs of the evaluators can be a good test for whether implicit requirements need to be expressed explicitly.

e. **Specification structure**. The evaluation should be functionally based, i.e. a comparison of how well the systems meet the functional and performance requirements for the different missions. A specification which is functionally structured will aid in evaluations.

While the specification structure and contents are important for the evaluation process, we consider that it is far more important that the specification is suitable for its other audiences, particularly the users and the developers. Any proposals for changes to specification practices to assist in evaluation should consider the effect on the other uses of the specification. It would be counterproductive to endanger the quality of the specification in the interests of evaluation efficiency.

4.7 How many requirements?

The number of requirements that a specification contains will be dictated by two factors: the level of the specification and the number of functions or features that the system is to implement. This section discusses the latter.

There is no "right" number of requirements. The number will vary considerably from project to project and is influenced by the application, the system size and complexity, and the risks. A minimum number of requirements are needed in order to adequately describe the users' basic requirements, sometimes called the core requirements. Additional requirements will add extra value in effectiveness and/or efficiency to the system. An attitude of "if in doubt, leave it in" increases the number of requirements, leading to accusations of "gold plating", "creeping featurism" or "requirements creep" [Manos 1993].

A project with unnecessary requirements can suffer a detrimental impact on integration, complexity, reliability, security, performance and cost, with little or no increase in usable functionality. These extra requirements can also divert attention away from the core requirements, diluting the effectiveness of their implementation and use.

A reduction in the number of requirements can be achieved by focusing upon the core performance attributes of the system. If a function does not have a critical impact on the performance of the system then it is important that the function is critically analysed with regard to its effects on risks, cost and benefit to the system as a whole. In other words, an assessment of the *value* of each requirement has to be made [Manos 1993]. Prioritisation of requirements is discussed further in section 6.5.

5. Specification checklist

5.1 General

In this section the attributes and qualities that contribute to making a good specification are examined. A checklist listing the attributes of good specifications is provided and some of the pitfalls of specification writing are discussed. Numerous references present various ideas on the desirable attributes and qualities of good requirements specifications. This checklist is based on Davis [1993a], Freedman and Weinberg [1990], IEEE-830 [1984], Jaffe et al. [1991], Kalms [1990], Leveson et al. [1994], Mar [1994], and Scharer [1990].

There is no doubt that few specifications for complex systems in the past can be regarded as "good". Our survey of industry's and projects' views showed this clearly. We believe, however, that most of the deficiencies in these specifications could have been detected much earlier by a combination of authoritative guidance and the application of experienced review at the correct times.

Table 4 illustrates the fundamental attributes of a good specification. Detailed comments on these attributes are provided in the remainder of this section. It should be noted that there is overlap and the potential for conflict between the different attributes. The overlap encourages the examination of specifications from different viewpoints. Conflicts, such as between readability and conciseness, must be assessed on the basis of the audiences of the specification, and tradeoffs should consider the risks of different approaches.

Table 4. *Attributes of good requirements specifications*

GETTING THE RIGHT REQUIREMENTS	
Correct, relevant	Is it what the user needs? Should it be in this specification?
Complete	Have *all* the requirements been addressed? Have the needs of all users been considered?
Problem oriented	Does it state *what* is needed rather than suggest *how* it might be implemented?
Contextual	Is the situation and environment for the use of the functions clearly stated? Are essential external interfaces identified?
Generic	Is what is asked for too specific? Is the need restricted only to this area?
ALLOWING ACCEPTABLE SOLUTIONS	
Feasible	Is it a realistic requirement? Is it cost effective?
Level of detail	Is the requirement too detailed? Is there enough detail to guide the designer?
Flexible	Will the requirement allow all acceptable solutions?
STATING THE REQUIREMENTS CORRECTLY	
Unambiguous	Will all readers understand what the requirement means?
Verifiable	How will the requirement be evaluated? How will the system be tested?
Readable	Is the specification readable by all the stakeholders and audiences?
Concise	Are requirements simply and clearly expressed? Are requirements duplicated?
Coherent	Are similar requirements collected together? If not, can you find related requirements easily?
GETTING THE SPECIFICATION RIGHT	
Consistent	Are there conflicts between related requirements? Is the terminology consistent? Is the same style and form of expression used throughout the specification?
Balanced	Is the level consistent across all areas? Are variations in level justified?
Organised	Is the structure logical and easy to follow? Can readers navigate the specification easily?
Modifiable	Will the specification be easy to modify?
Traceable	Can each individual requirement be referred to easily?
Well presented	Is the format appropriate? Does the use of language and grammar enhance the quality of the specification?

5.2 Getting the right requirements

To maximise the likelihood of the system meeting the users' needs, the right requirements must be defined.

a. **Correct, relevant.** Requirements must correctly reflect the genuine needs of the users, and they should be focused on the system being specified and functions and performance needed.

Ensuring the correctness of requirements is difficult. It will be enhanced by adequate requirements capture and validation (sections 4.3 and 4.4) activities, and supported by tracing requirements to higher level requirements (section 4.5).

Requirements which should not be in RFT specifications, but if considered necessary should be elsewhere in the RFT package, include [Kalms 1990]:

- Contractual issues such as price, schedule, warranties, rights and ownership of intellectual property
- Contract management issues including project milestones, meetings, reviews, delivery (although it may be necessary to discuss phases in terms of the functionality to be provided in each phase).
- Evaluation criteria and method.

While these should not be included in a specification as *requirements*, it is often useful to include other information relevant to some of these issues, particularly in early versions of the specification. Providing information on how a function might be evaluated or tested will in some cases be easier and more logical than defining the requirement in further detail. Where this information is critical, its presence in the RFT specification may serve as a pointer to other documents where it is defined more prescriptively. For example, indicating that symbology or the layout of screen forms will be subject to Navy's review and approval highlights the fact that Navy intends to approve the design in this area (design approval is not normally a customer responsibility in Navy projects). While this is a contract management issue, it provides useful guidance to tenderers which might otherwise be overlooked.

Including such information in early versions of the specification can also serve as a reminder that these issues need to be included in the RFT documentation, before the appropriate RFT sections have been drafted.

b. **Complete**. It is important to define a complete set of requirements, reflecting all the needs of the users, and the needs of all users. Detecting missing requirements is difficult but the risks resulting from not doing so can be very high, both in cost and performance terms. Once the specification is agreed, omissions are likely to be found only by accident, usually much later in the project life cycle. One reason for this is that the project team will tend to focus on the defined requirements in the specification, rather than the genuine requirements.

If the system is complex or some of the requirements are not known or vague, a useful strategy may be to first specify a "core" set of requirements defining critical functionality at a relatively high level, deferring the other requirements to a later release. This allows the requirements team to focus on the breadth of the requirements, without being distracted by problems of detail. There are risks with this strategy also however,

because there will be a temptation to defer requirements which have not been analysed in detail or are difficult to specify to a later release when they really need to be defined early.

A specification is not complete if it contains implicit requirements which may not be inferred by all typical readers (see section 4.1). Trying to eliminate *all* implicit requirements is an impossible task - readers must be credited with some understanding of the application and engineering practices - and trying to do so is likely to lead to a grotesque and unreadable document. It is also possible that excessive paranoia in this area may lead to additional ambiguities as superfluous requirements are added.

c. **Problem oriented.** The requirements specification should specify the problem and avoid suggesting or presupposing the solution. It should generally be free of design details and specific product information.

Some degree of assumption of the design is unavoidable. For example, some would argue that to assume a database will be used in a specific application is presupposing the design. In most cases however, such an assumption will be reasonable and significantly improve the readability of the specification.

Care must be taken to ensure that other viable options are not inadvertently precluded. One method of doing this is illustrated in the following example. The recording requirements for a system were based on using tapes. The requirements included the capacity of the tapes (in terms of level of recording detail and mission hours), the minimum time to read information back into the system, and the need for industry standard tape types and recording format. These were valid requirements which would have been less readable, and more difficult to specify, if they were worded to apply to any recording medium. Prior to these requirements was a statement that any recording medium would be considered which effectively met the requirements following. In fact, most tenderers proposed opto-magnetic disks, a new technology, and the method of specification caused no problems for either the tenderers or the evaluators.

Using specific products as examples may also be acceptable under some circumstances [Kalms 1990; *Specification Handbook* 1988]. For example, specifying the individual functions of a wordprocessor may be counterproductive when there are one or more well known products which meet the requirements. It is important that it is stressed that the products are specified only as examples, that they are supported by valid functional requirements, and that other products will be given equal consideration. It is preferable to specify at least two different products in such a situation. This method of specification is obviously open to abuse, and its use is recommended only in cases where there is no other effective means of expressing the requirements clearly.

There is always a danger when requirements address solutions that the drafters will concentrate more on the solution than the problem. The inevitable result will be distortions and omissions in the requirements, which will be detrimental to the resultant system.

d. **Contextual.** Specification drafters must consider not only the system and its operation but the context in which the system will be used.

The operating context will include the external environment (e.g. sea states, wind, temperature), the equipment environment (e.g. temperature, humidity, vibration) and the operators' environment (e.g. temperature, distractions, stress).

Interfaces and interactions with external systems must also be considered and where they are unchangeable, need to be specified as completely as possible. Examples include data links, exchange of computer data using tapes or disks, message formats and control signals. Interfaces are one of the most critical sources of risks in complex system development, and cannot be treated casually.

e. **Generic**. It is often difficult to decide whether a specific or generic (multi-purpose) solution is required. For example, a message handling system may only need to handle a few pre-defined messages, but it is known that the system may need to be extended, even in development, to handle other messages. This decision needs to be made during the requirements phase, because the needs of the users will ultimately drive the design. If the users have a need for a generic solution, or one which may be easily extended, this must be stated.

Specific solutions are often cheaper than generic solutions and may result in a simpler user interface. The following issues should be considered before deciding between a specific or generic approach. Specific requirements:

- Are not resilient to change - they will be difficult and expensive to modify.
- Are easier to get wrong because they are at a lower level of detail - this has been a common cause of project failure in the past.
- Require more certainty about detailed needs early in the project - generic requirements can be an interim defence against uncertainty.

5.3 Allowing acceptable solutions

Requirements must be expressed in a way that maximises the probability that all acceptable solutions can be accommodated.

a. **Feasible.** Requirements should be achievable within the constraints of time, cost, and the available technology - they should be realistic.

Specifying unrealistic requirements not only adds risks to the project, but can also result in an overall lower level of system performance than would otherwise be achievable. This occurs because when it is realised that the

requirements are not achievable, the design is often well advanced, and accommodating less demanding requirements in an optimal fashion may not be possible without an unacceptable (in terms of cost and schedule) major redesign of the system. In addition, formulating the revised requirements typically occurs in an atmosphere of crisis, which is not conducive to reflective requirements analysis.

Making a decision on whether requirements are feasible or not is a difficult decision for the users. It needs competent and authoritative advice from materiel experts, and often further investigation and analysis of what is feasible. The inclusion of personnel with an understanding of the potential solution area will assist in this regard. The decision is not made easier by extravagant and misleading claims from potential system suppliers which can obscure the maturity and performance of products and technologies. Users are understandably reluctant to accept reduced performance when advanced solutions appear to be tantalisingly close.

Long project schedules contribute to the dilemma. It is often said that "there may be no solution now, but there might be by the time the system is designed". It needs to be realised however that most decisions on equipment selection and high level design are made during the preparation of the tender, or at latest only a few months after the contract is let. Changing the design at a later stage will normally incur cost and schedule penalties, and will be strongly resisted by both the contractor and the Project Authority.

b. **Level of detail.** Requirements should be written at a level of detail that allows developers to produce a design that will satisfy users' requirements. Too much detail will overly restrict the scope of a developer's choice of a solution (and possibly increase cost and schedule), whereas too little may result in a compliant but unacceptable solution. This is discussed further in section 6.3.

c. **Flexible.** Closely related to finding the right level and being problem oriented, flexibility is needed in specifications to ensure that all acceptable solutions are encompassed. Requirements should not restrict the possible solutions to those which are known to the drafters. This applies both to products, where there are known products which meet the users' needs, and to technologies, where there may be emerging technologies which are more cost effective than those implied by the specification.

5.4 Stating the requirements correctly

The written requirements must be clear and not be open to misinterpretation.

a. **Unambiguous.** At best ambiguous requirements will cause confusion; at worst they will lead to the wrong solution. All readers must have a common interpretation of what each requirement means. Checks should be made to consider different interpretations and eliminate them. Section

6.8, for example, shows how injudicious use of language can result in ambiguity.

It is important not to become paranoid about ambiguity. The attitude of "if one reader misinterprets it incorrectly, it must be badly worded" is not necessarily correct. There will always be the possibility of a reader misreading a requirement and on proper reflection understanding its intended meaning, regardless of how well it is worded. Trying to avoid this possibility can result in other problems, particularly in the overall readability of the specification. Rather than change the requirement itself, it may be preferable to add a clarifying comment (see d. below).

One common source of ambiguity and conflicts between requirements is the presentation of different viewpoints, particularly by different drafters, which may relate to or embrace the same requirements. An example is different navigation requirements in different missions or roles. In such a case it may be prudent to address navigation in its own section, and provide cross references from sections which refer to navigation.

b. **Verifiable.** There is little point in specifying requirements which cannot be evaluated or verified. Each requirement should be checked for its ability to be verified. Where the method of verification is not self evident, or there are different degrees of verification which might be applied (in assessing reliability, for example), some indication might be given of how the requirement is to be evaluated or tested. Verification methods can involve testing, demonstration, analysis, inspection and comparison with other systems.

It should be noted that in Navy projects it is the contractor's responsibility to show that the requirements are met. Provision of guidance can only indicate a level or method of verification which may be acceptable to Navy.

In an RFT specification, some requirements will not propagate into the contractual specification. Their purpose is to stimulate proposals from the tenderers which can be evaluated. In these cases, the full performance, or the product, will be agreed during negotiation. The requirement for a wordprocessor, for example, might be replaced by a specific wordprocessor. Usability requirements might be replaced by requirements defining a user interface philosophy.

Requirements including words such as "typically", "generally", "usually" or "normally" (such as in "this function shall normally be available ...") cannot be tested unless the conditions for compliance are more clearly defined.

c. **Readable.** Although the target audience for the specification is the system developer, it is important that the specification is readable by all audiences including the Project Authority and the user representatives (see section 3). If readers consider the language, structure and form of expression to

be unfriendly, they will also have difficulty in understanding the requirements.

d. **Concise.** Requirements should be written using the simplest, clearest language possible. All information which is not needed to understand the requirement and respond to it should be excluded. Conciseness also means reducing duplication and redundancy and in so doing should minimise ambiguities. Care must be taken here because to some extent conciseness can reduce the readability of the document, although in general it will improve it.

Where information is included to clarify requirements, it is important that it is clearly indicated as such (e.g. by including all such information in italics, prefixed by the word "Guidance"), and does not conflict with other requirements or other clarifying information provided elsewhere. It is a common error for specification drafters to treat supporting information with less respect than requirements, leading to ambiguities or conflicts.

e. **Coherent.** Related requirements should be gathered under the one section in the same specification, using cross references where necessary. Fragmentation of requirements reduces the readability of the document and can lead to conflicts and omissions, particularly when requirements are changed.

Coherency in complex systems is often difficult to achieve, due to the numerous interrelationships between functions. Making a decision on whether to group requirements in a single section should include the following considerations:

- Will readers know where to look to find these requirements? Will they realise that there are similar requirements elsewhere?
- If the requirements change, will it be obvious what needs to be changed in the specification?
- Is the requirement in the right section, particularly if it is referred to elsewhere?

5.5 Getting the specification right

The specification as a whole must present the requirements in a logical, consistent manner.

a. **Consistent.** It is essential that requirements do not conflict with each other, explicitly or implicitly. Any conflict will result in uncertainty in what the genuine requirement is, and may make the evaluation or verification of requirements very difficult.

Similar requirements need to be expressed in the same way. Variations in the way requirements are expressed may result in tenderers assuming there is some underlying reason for doing so, resulting in their misinterpreting the requirement.

Terminology in the document should be used consistently. Where there is a possibility that some readers may not be familiar with the terminology, it should be defined, using a widely accepted definition.

b. **Balanced.** Different requirements of differing complexity may have to be specified at varying levels of detail (see section 6.3), but as far as is practicable the document should be balanced with a consistency of level.

c. **Organised.** The specification should be organised in a logical fashion using a structure that is easy to follow and facilitates navigation through the document.

d. **Modifiable.** The structure, coherence and traceability of the specification need to facilitate modification and maintenance. In general, a specification is regarded as easy to modify if the relevant requirements are easy to find, and the effect of the changes on other requirements can be readily established.

e. **Traceable.** A consistent numbering scheme, with a unique number identifying each individual requirement, will enable the requirements in the specification to be easily traced to other documents (such as the tendered technical proposal). It is not necessary to provide sequential numbers for each requirement, or if this is done, for adjacent requirements to have adjacent numbers. The main objective is to uniquely identify each requirement. This clause, for example, is uniquely referred to by the number "5.5.e". Partitioning of requirements is discussed further in section 6.6.

f. **Well presented.** Use of good consistent grammar, spelling and format will enhance the readability of the specification, reduce ambiguity, and generally provide confidence that the specification is a serious and considered statement of requirement.

6. Detailed specification development issues

The specification checklist in the previous section can be used as a straightforward device to check the quality of a specification. In this section, some of the more important specification issues are examined in more detail. Many of these arose in our industry and project surveys, described in Reports 1 and 2 respectively.

6.1 Difficulties in writing specifications

Some specifications are easier to write than others, but it is generally accepted that specification drafting is not an easy task [Zave 1990; Scharer 1990]. The following factors affect the level of difficulty in drafting the specification.

a. **Understanding of the users' needs.** If the system is new, potential users will often have only a broad understanding of their needs. Similarly if

there are no obvious "users" at the time of drafting, it will be extremely difficult to identify correct and complete requirements. It is also important that the right users are available and that they can reach consensus (see section 4.3).

b. **System type.** Systems supporting human analytical processes (e.g. executive information systems, C3I systems) are more difficult to specify than systems with little or no human interaction requirements (e.g. control systems, propulsion systems). Specifying human computer interfaces and the performance of operator intensive systems in terms that can be tested is not always possible. Sometimes the only solution to this problem is to recognise that the specification is vague and use prototyping or some other iterative development methodology to validate the requirements.

c. **System size.** The larger the system the greater the volume of the accompanying documentation. This can result in increased opportunities to omit information from the specification and miss inconsistencies, purely because of the bulk of the information.

d. **System complexity.** Large systems tend to be complex. Complex systems are difficult to specify because of the number of interrelationships between functions and subsystems. Requirements may also be spread or duplicated across several subsystems so that simple partitioning of requirements may not always be possible. The risk of inconsistencies increases and many of the attributes of good specifications such as balance, completeness, non-ambiguity, and correctness are all much harder to achieve.

e. **Similarity to existing systems.** If a similar system exists, the specification will be easier to write, not only because specifications may exist for the previous system, but also because there is much more experience in its use and design. However, the temptation to liberally copy requirements from a previous specification should be curbed in most cases, because this may overly restrict the development of requirements for the new system, or force its design to use obsolete techniques and technologies.

f. **Resources.** If the drafting team does not have the correct resources either in the number or quality of personnel, this will adversely affect the quality of the resultant specification (see section 8.1).

6.2 Content and completeness

There are no hard and fast rules about what and what not to put in a specification. Anything that contributes in assisting the various audiences in understanding the requirement, but does not detract from the quality of the specification, is acceptable. Completeness can only be thoroughly tested by requirements validation techniques (see section 4.4).

6.2.1 Assumed knowledge

When most of the drafters have a good knowledge of the operational requirements, a common error is to assume that all readers of the specification share this knowledge. This can result in misunderstanding of the specification by tenderers. On method of testing for unjustified assumptions is to have at least one member of the specification review team who has little knowledge of the operational requirements.

6.2.2 Copying requirements from previous projects

Considering requirements from previous projects is often a useful method of compensating for lack of drafter experience. It can often complement the requirements capture process. Direct copying of requirements from earlier projects is not recommended, however, for the following reasons:

a. Requirements which are suitable for one project are rarely suitable for another. The roles of two ships may be different, for example, and the required performance of their systems will usually be different. Each system has its own requirements which need to be considered without being influenced by the biases of another.

b. Requirements copied from an older project may not reflect advances in tactics, technology or system performance.

c. There is rarely any consideration of the quality of the original requirements. Copying them may amount to copying the errors of the past.

6.2.3 Copying requirements from brochures

Although promotional material ("glossy brochures") provided by defence suppliers is a useful source of information, drafters should be very wary about using the information directly in specifications for the following reasons:

a. Figures provided in promotional material are often the best performance that can be achieved under optimal environmental conditions. They are almost universally more optimistic than the typical performance which might be expected from systems, or the performance that a tenderer can guarantee in a contract. Use of such figures will ensure that no tenderer can be compliant.

b. Where several of the performance characteristics of a specific product are echoed in a specification, the specification may reflect the actual design of that system rather than Navy's higher level requirements. Competing systems will have a different set of characteristics which, while they may meet the higher level requirements, cannot match the specified requirements point by point.

c. Using such figures does not guarantee feasibility. In some cases the equipment or software described does not exist, or is in an early stage of development.

The likely consequences of injudicious use of performance figures from promotional material are that competition is decreased and that there is less likelihood that existing equipment will meet the requirements.

6.2.4 Technology constraints

Particular care should be taken when requirements relate to immature, emergent or rapidly changing technologies. In these cases requirements should be stated at as high a level as is prudent. This is particularly relevant for long projects where technology advances during the course of the project may make some detailed requirements (such as requirements for computer types and capacities) either obsolete or difficult to achieve.

6.2.5 Use of standards

The referencing of military specifications and other standards requires careful review during specification development:

a. Referenced standards must be appropriate for the application.

b. All standards listed must be referenced in the text of the specification.

c. Some standards (eg DOD-STD-2167A) require specific decisions to be made or actions to be performed by the customer. It is important that drafters and the Project Authority are aware of these obligations.

d. Some standards (eg DOD-STD-2167A) require tailoring to correct conflicts with other standards or to meet specific project objectives. Tailoring might include the reduction in required documentation or the provision of specific information not catered for in the standard. This information may be included in the specification or a separate document. Process standards such as DOD-STD-2167A may also need additional tailoring in conjunction with the developer after selection of the preferred contractor, to ensure compatibility with the development methodology proposed.

e. Many standards are multi-level providing for different levels of compliance depending on the needs of the application. In these cases the particular applicability of the standard should be reviewed and stated.

f. The use of standards which are not widely used either in defence or industry, or are obsolete, can seriously impact project cost and schedule. Drafters should ensure that the standards referenced are current, valid and available.

With regard to standards it is recommended that specific separate justification be given for the use of each standard referenced, including a mandatory review of the currency, applicability and preference over alternative, including civil,

standards. The report should also address alternatives within the standard, if any, and actions which the standard places on the Project Authority.

It is also recommended that Navy institutes a comprehensive review process to upgrade or retire relevant standards, and provide guidance on the use of specific standards.

6.2.6 Partial requirements

It is often necessary to define requirements which are partially complete. This might occur in expressing the need for a wordprocessor, for example, where a general purpose COTS product is required, but where there are also specific needs for the application. It would be difficult and unnecessary to define all the functions needed. In this type of situation it is acceptable to specify both general requirements (e.g. "A general purpose wordprocessor shall be provided") and inclusive specific requirements (e.g. "Specific functions shall include the following ...").

It is important that the general and specific requirements are clearly separated in different clauses, however, to indicate that they are separate individual requirements, which must be met individually. This will assist both tenderers and evaluators in interpreting the specification. If the requirements are stated in the form "A general purpose wordprocessor shall be provided, which includes the following functions ...", there is reason to believe that any wordprocessor with only the nominated functions will be acceptable.

6.3 Level of requirements

The level to which requirements are specified will vary according to the type of system being developed, the level of experience of the drafters, and the perceived level of experience of the developers. Specification drafters depend heavily on their own and Navy's experience both in the application domain and technologies involved. Developers with extensive experience in a particular domain prefer high level specifications, whereas the same tenderers will prefer detailed requirements in areas where their experience is less. Most problems in level of detail, both too high and too low, occur when the drafter has insufficient or inappropriate experience. The most serious problems occur when both Navy and the developer have a low level of experience.

Ideally the specification will be written at a level providing enough information for the developer to produce a cost effective design, but which allows freedom for the developer to choose between alternative solutions.

Overspecification, where the level is too low, can lead to:

- Solutions which are exactly compliant with the low level requirements, but which do not meet the genuine requirements for the system.
- Solutions which cost more than they should.
- Reduction of the number of tenderers who may bid for the development of the system, because the requirements constrain solution to a limited number of products.

- The development of parts of the system rather than the use of non-developmental items.

Underspecification, when the level is too high or the requirements too vague, also involves risks. The major risk is that the users may not find a system acceptable, even though it is technically compliant with the high level written requirements. Most developers also want the system to meet the users' needs, so that they also are concerned with this risk, as shown in our industry survey.

It is evident that the choice of specification level must be based on risk. Requirements should be specified at the highest level at which the risk is acceptable (see section 3.5).

In some cases it is very difficult to specify a requirement in other than vague terms. The need for a "user friendly" interface is a genuine requirement, but "user friendliness" is very difficult to objectively evaluate or verify. Specifying this in more detail may not improve the specification significantly, but could increase the probability of constraining the design unnecessarily. In these cases it is suggested that either the requirement should be clarified if possible, or that guidance should be provided on how tenderers should respond to the requirement, i.e. what information they should provide in the tender to assist in evaluation.

In recent Defence projects there has been strong resistance by some contractors to include derived requirements in the acceptance testing of the system. Instead, the system is accepted against relatively high level performance based requirements (see also section 3.6). While this superficially ensures that those requirements are met, it does not test *how* the requirement is met, or *how well.* Limited visibility and control by the Project Authority of integration and internal system testing increases this risk. If these special circumstances are likely to apply, it will be necessary for the customer to specify all functions that should be tested during the contractual system acceptance.

6.4 Structure and format

Structure here refers to the methodical approach to organisation of the requirements in the specification. Format refers to the layout of the specification document, which can also dictate or constrain the structure. It is currently Navy policy that RFT specifications should follow the structure defined for a Top Level Specification (see section 1.5).

The use of a logical, consistent and defined structure has the following advantages:

- It gives all individual drafters an understanding of the scope of their area of responsibility and its relationship to other sections [Rushforth et al. 1990].
- It ensures that there is a place in the structure for most common requirements, which will reduce omissions.
- It will help maintain consistency by reducing overlap between sections [Kalms 1990].

- It makes information easier to locate (with a good table of contents).
- It assists in tracing requirements.

Requirements specifications are usually developed and provided in electronic form to developers who parse the requirements into their own tools and databases. In the parsing process the specification is separated into individual requirements, with clarification and categorisation information added during the process. As shown in our industry survey, various characteristics of the specification structure can assist or detract from the parsing process. An example of this is the need to separate and uniquely identify individual requirements (see section 5.5).

Our industry survey noted objections to certain practices in the layout of specifications which make parsing more difficult. In general we support the use of these practices where they are used to increase readability. It should be noted that readability is the key priority in allowing stakeholders to contribute to and improve the quality of specifications, and this should be preferred over a relatively mechanical issue such as automatic parsing, which occurs infrequently. The following practices were identified as a barrier to efficient parsing:

a. **Use of tables**. Tables are often a useful way of presenting large numbers of simple requirements such as data items. Tables should be minimised, however, except in cases where the number of items is large and the requirements can be presented in an effective way.

b. **Use of lists.** Lists (such as this one) also offer an effective and readable method of listing requirements which have a common scope. We consider that no restrictions should be put on the use of lists.

c. **"Overlap" requirements.** In this case the requirements for one section refer to the requirements in another, with specific additions or exclusions. An example of such a requirement is "all data in paragraph 4.17 shall be recorded, with the exception of operator errors and BITE data". We consider that this form of specification is inherently risky - if a requirement in the original section is changed or added, there is little indication that the list is referred to in another section, possibly resulting in an error in requirements.

   The obvious alternative to this is equally risky. By listing each item of data in each area that refers to several items, there is a real risk that the different data item definitions will not be coordinated. This also ignores the objectives of grouping related requirements together and avoiding duplication. Two other approaches might be considered:

   - Using an object oriented approach for data related requirements, showing in the definition of requirements for each data item whether it needs to be recorded, displayed etc. This could also result in serious duplication, however.
   - Defining the data requirements in a section dedicated to that purpose, and referring to each data item by name, but with no other qualifying information apart from a cross reference to the data

requirements section. The data section should include cross references to all sections which refer to it, to aid in safe modification.

d. **Umbrella clauses.** In an umbrella or "catch-all" clause, the scope of a requirement applies to all or several system components. Typical umbrella requirements refer to build constraints or user interfaces. Objections to such requirements stem from the fact that although it is obvious to tenderers that the requirements cannot be applicable to all system components, particularly non-developmental items, there is uncertainty in how far the requirements extend to modified items, for example, or how they should respond with regard to non-developmental items.

   The use of such clauses obviously reduces duplication and increases readability. It is evident, however, that the scope of such clauses needs to be carefully considered and specified, if it is not intended that the requirement should apply to the whole system. This may be done by indicating criteria by which the application of the requirement might be assessed, or by limiting the requirements to specific components, if these are known at the time of specification.

e. **Conditional requirements.** Requirements may only apply under certain conditions. In such cases it is important that the conditions are explicitly defined and that the tenderer understands the conditions and can detect when they are valid. An example for a point defence system may be "The warning shall continue while the missile is a threat to the ship". While this is a genuine requirement, it presupposes the ability of the system to determine whether or not the missile is a threat. If the specification drafters can confidently clarify this requirement in terms of what constitutes a threat, they should. Otherwise the users will need to accept the developer's algorithm for threat evaluation.

6.5 Prioritisation of requirements

In most specifications prioritisation is limited to mandatory ("shall") and non-mandatory ("should" or "desirable") requirements. This separates essential from non-essential requirements, but provides no indication of the relative priority of requirements within these two classes.

Prioritisation of requirements into more levels of priority would be useful for both tenderers and materiel personnel. In our industry survey participants felt that a clear prioritisation of requirements would improve their ability to apply cost benefit analysis and criticality analysis techniques in Navy projects and improve their ability to offer cost effective solutions. It was suggested that prioritisation to 3 levels (e.g. mandatory, highly desirable and desirable) would be a marked improvement on current practices. Where prioritisation is not provided, clear definition of operational requirements, including the operational concept, would help tenderers to assess priorities themselves (see also section 7.1)

Materiel personnel would find the prioritisation useful in developing evaluation criteria and weightings, and in contributing to tradeoff studies where necessary.

It is often difficult to assign priorities to requirements with confidence that those priorities are agreed, correct and stable, i.e. that will not change even over a short period. This may account for the reluctance to do so. There are various well known and proven techniques for assigning priorities to requirements, including Quality Function Deployment [LaSala 1994], Analytical Hierarchy Processing [Golden et al. 1989] and decision trees. There are also tools which can help in this regard as shown in section 8.2.

6.6 Precedence of different requirements

The value of good requirements can be degraded or even lost if the precedence of different requirements is not defined, or is defined incorrectly. The level of precedence establishes which requirement (or specification) will be considered to be contractually valid in the event of conflicts between two or more requirements being detected. Precedence is different from priority (see above) although the two are related, and provides a mechanism for handling conflicts between requirements which address similar or related subjects. Precedence usually only becomes an issue later in a project when there may be disagreement between the Project Authority and the contractor as to whether certain performance and functionality requirements are contractually defined.

Generally the precedence of different specifications needs to be stated, usually with higher level specifications taking precedence over lower specifications. Specifications at the same level need to be examined on a case by case basis. The precedence between the specifications and other material in the RFT package should also be determined and clearly stated. If the precedence is not stated, default precedence will apply, where it may be assumed that requirements specified first (in a series of specifications or within a specification) have higher precedence than those that follow.

Where a specification contains requirements in major sections which are at different levels, such as happens when the operational requirements are included for information early in the specification, the precedence between the different sections also should be stated. If there is confidence that the detailed requirements completely encompass the operational requirements, it may be preferable to state the operational requirements as objectives rather than requirements, to avoid the need to resolve conflicts and simplify the evaluation process (see section 4.6).

6.7 Grouping and partitioning requirements

The grouping and partitioning of requirements refers to the manner in which requirements are clustered and separated both within the specification and across different specifications, and the means used to separate different requirements. The different issues are as follows:

a. **Grouping of related requirements.** Where possible, requirements should be grouped according to their function. The grouping of related requirements together also adds to the readability and structure of the specification.

b. **Separation of requirements.** Each clause in the specification should contain only one requirement. This not only assists Navy and tenderers to parse and trace requirements; it also encourages structure in the specification and discourages the merging of unrelated requirements. It also clarifies the scope of each requirement.

What an "individual" requirement consists of is a subjective issue which will always be a matter of argument, as shown in our industry survey. We suggest that requirements be separated as far as is practical, without seriously affecting readability. For example, we would regard the requirement "The target's bearing, range, course and speed shall be displayed" as a single requirement, even though it strictly contains 4 requirements.

c. **Numbering of requirements.** A consistent numbering scheme with a number allocated to each individual requirement will enable the requirements in the specification to be easily referred to and traced to other documents (such as the tendered technical proposal). The main objective is to uniquely identify each requirement. This clause, for example, is uniquely referred to within this document by the number "6.6.c".

We consider that it is not necessary to provide sequential numbers for each requirement, effectively numbering each "shall" or "should", although this has been suggested by some authors, and by some tenderers. While there may be some advantages in such an approach there are several drawbacks:

- The numbering interspersed with the text will reduce readability.
- Managing the numbers can be difficult through many changes of the specification.
- There is likely to be confusion between clause and requirement numbering.
- It is much easier to test that each clause is numbered, than to test that each individual requirement is numbered.

In any case, a consistent policy and practice of "one requirement per clause" should make this alternative method of numbering redundant.

d. **Separating functional and non-functional requirements.** It is generally recommended that functional and performance requirements should be separated from non-functional requirements. The draft Top Level Specification format (see section 3.2) supports this approach.

e. **Separating functional from performance requirements.** Some references support the separation of the requirements for functions and performance into separate sections (including Kalms [1990]). We consider that in most

cases performance requirements are closely related to functional requirements (see section 2), and that this practice would result in fragmentation of the requirements and reduced readability.

f. **Separating mandatory and non-mandatory requirements.** Another partitioning strategy involves separating mandatory ("shall") from non-mandatory ("should") requirements. Tenderers look favourably on this strategy because it allows them to concentrate their efforts on the essential requirements on which their tender response is predominantly based.

We consider that this would also result in excessive fragmentation of the requirements. This view is supported by a study [Rushforth et al. 1990] into specification practices of the CCTA organisation, the UK Government Centre for Information Systems, which had previously followed a policy of separating "shoulds" and "shalls". The study found that the practice had led to numerous errors in specification (particularly conflicts and ambiguities, which are symptomatic of requirements fragmentation) and recommended that the practice should cease.

Mandatory and non-mandatory requirements should not be included in the same clause, however, and when there is a mixture of "shalls" and "shoulds" in a list, it is preferable that the non-mandatory requirements are listed after the mandatory requirements.

6.8 Use of language

The language used in the writing of specifications should be clear and concise. It is usually preferable to use a semi-formal style than to use a more descriptive, conversational or variable style and risk introducing errors and ambiguities. Jargon should be minimised except where it is likely to be understandable by all audiences. Where specialised terms are used, they should be defined in a glossary and used in a consistent manner.

6.8.1 Special and dangerous words

Some words such as "shall", "must", "should", "will", "desirable", and, "or" have special meanings when used in the context of a requirements specification. Misuse of these words will cause confusion between contractors and procurement authorities. Redefinition of these words may also cause problems. Navy should standardise on a consistent definition to be used in all projects.

Other words such as "some", "sometimes", "often", and "usually" are vague and ambiguous. These words can result in requirements being unverifiable and should be used only with great care. Freedman and Weinberg [1990] provide a comprehensive list of words which are potentially dangerous.

6.8.2 Alternative functions

Where alternative functions are specified, the wording should clearly show whether the choice is being offered to the tenderer or the operator. An example might be "Positions shall be entered either graphically or numerically using the keyboard". This probably means that the operator needs two alternative methods of entering position, but it could also be interpreted as meaning that the designer has a choice of providing either graphical or numeric entry, without providing both. The requirement "Facilities shall be provided for the operator to enter position information both graphically, and numerically using the keyboard" is less open to misinterpretation. The use of "or" in a requirements clause can quite often be ambiguous.

6.8.3 Problems with numbers.

The specification of minimum and maximum values or numbers can sometimes result in the exact opposite of what is intended. Consider the following requirement: "The system shall process a maximum number of 200 tracks", which is often also worded as "The system shall process up to 200 tracks". The requirement here is that the system should have a *minimum* capacity for 200 tracks (which should have been stated), but the drafter is also recognising the fact that in many cases fewer will need to be processed. Instead, a system with a capacity with 10 tracks would, superficially at least, meet the requirement.

Another way of expressing this type of requirement, which has been used in Navy specifications, is "The system shall process up to a minimum of 200 tracks". The requirement now is becoming ambiguous, because two separate ideas are being combined in the same requirement.

Care also needs to be taken with quantities such as precision, accuracy and scale. We *talk* of accuracy, for example, but often *specify* in terms of errors, and accuracy increases as the error decreases. Consider "The position displayed shall have a minimum accuracy of 10m". This could be misinterpreted, to the customer's disadvantage, as an accuracy of no better than 10m, i.e. 10m is the minimum error. While it might be argued that the requirement is clear, there is sufficient ambiguity for the developer to argue otherwise, which has happened in several instances.

In these cases, there are two acceptable solutions:

- Use the words "better" or "worse" rather than maximum and minimum, e.g. "The position shall have an accuracy better than 10m".
- Express performance requirements only in terms of the values used, e.g. "The position shall have a maximum error of 10m (error 95%)".

These examples show the need to carefully examine the language used, and to minimise the risks of misinterpretation.

6.9 Difficult areas to specify

Apart from non-functional requirements, which are discussed in the next section, there are several other areas which are difficult to specify. Some of these are discussed in this section.

6.9.1 Concurrency

Whether or not functions are required concurrently is often not clearly specified. This often results in problems when the contractor designs functions to be mutually exclusive but the customer assumes that the functions will be available simultaneously. For example, if an operator requests a hard copy plot of the current situation, it is unlikely that suspension of other functions at that console would be tolerable for 5 minutes (say) while the plot is compiled and plotted. This type of situation needs to be anticipated and accommodated in the requirements.

6.9.2 Expression of statistical performance

Often performance cannot be tied down to single number or range and should to be specified in statistical terms. An example is navigation accuracy where specifying a 2 dimensional positional accuracy of ±25m has little real meaning. More importantly, it will be interpreted by many tenderers as meaning 25m (1σ) or 25m (CEP) which will mean that more that 30% of all errors are greater than 25m. This is almost certainly more tolerant than was intended by the users. There are numerous ways of stating the requirements in statistical terms, but without clear definition of what the terms mean (σ or "standard deviation" could be calculated in more than one way for a 2 dimensional error) the performance will be at risk.

We recommend the following definition for statistical performance figures. It has the advantages of being meaningful to the users and developers, and of being simple to test.

> **Error 95%**: Refers to the tolerance of a measured or calculated value. 95% of all observations result in an error less than or equal to the specified tolerance. For positional values, the error is the radial distance from the true position.

The above requirement would then be stated to be a position accuracy of 25m (error 95%). It would still be necessary in the testing of this accuracy to establish the test conditions and the number of observations. If these are likely to be disputed, they may need to be included for information in the specification.

6.9.3 Existing products - COTS and NDI components

Commercial off the shelf (COTS) components or non-developmental items (NDIs) are often preferred because of a combination of proven performance and

reliability, reductions in development risk, use of modern technology, shorter acquisition times and lower cost [Rhoads 1992].

Before examining how systems containing such components might be specified it is useful to look at the benefits and risks in using such components. Kirkpatrick et al. [1992] identify the use of COTS as one of four major risk areas in developing complex software intensive systems. They identify the following factors as adding to product risk:

a. **Customising.** Although the use of a COTS component is often represented as "plug and play", it has often been found necessary either during development or after acceptance to modify the component to meet the users' needs or to interface with other components. This can often lead to serious risks because of the need of the contractor to modify software which was probably developed elsewhere, or to bring another subcontractor into the project - the COTS developer.

b. **Testability and integrability.** If a COTS component fails a test or does not integrate well with other components, there is often little leverage available to the contractor or the customer to get the COTS supplier to repair it. The impact of any defect is therefore magnified because of the lack of an effective repair mechanism. The fact that most commercial software products offer no warranty of performance contributes to this problem.

c. **Quality issues.** In most cases the performance or reliability figures for COTS components are not known. This can make estimation of system performance or reliability figures very difficult. As Kirkpatrick et al. state: "the system requirement is dependent on a key component whose actual performance is unknown and unalterable".

Similar problems can apply to NDIs, where components developed for other systems are proposed for a new system. In this case, of course, the NDIs are unlikely to have the other benefits of COTS components, due to their lack of widespread use.

Specifying requirements when it is known that many contenders will offer solutions including COTS components or NDIs raises a dilemma. Specifying at a high level will increase the performance risk if a component needs to be developed (see section 3.6). Specifying at a level suitable for development will often result in some existing components being non-compliant. While the component is strictly non-compliant, it may offer benefits as detailed above as well as additional functionality, which can be traded off for the shortfall in specified functions or performance.

One approach is to make the lower level requirements non-mandatory in cases where COTS components or NDIs are likely to be acceptable. This may result in specification risk, however, if the component needs to be developed, which must be resolved prior to the signing of the development contract.

The problem is probably better addressed at the contractual level with formal advice being provided to tenderers that existing equipment which is substantially compliant will be preferred over developed or redeveloped components, if sufficient benefits for its selection can be clearly demonstrated.

It is also important that the main barriers in the past to the use of COTS, namely the extensive use of military standards for components, are carefully reviewed so that they do not exclude acceptable components [Rhoads 1992]. Rhoads also stresses that changing the acquisition process to increase the acceptability of COTS and NDIs requires:

> ... earlier consideration of lower level detailed requirements and additional planning during the beginning of the acquisition process. These changes in themselves could delay an acquisition when an operational need is not well defined initially.

6.9.4 Integration

The level of integration of system functions was identified in our surveys as a problem area. It is extremely difficult to define levels of integration. Requiring "a high level of integration of functions" is recognised as a genuine statement of need, and it can be reasonably evaluated on a relative basis, but it does not provide sufficient guidance to the tenderers such that they can be confident that their tender will be compliant in this area.

In addition, a high level of integration may cause problems in the implementation of a solution. Increasing the level of integration:

- May increase the single points of failure.
- Will almost certainly increase the complexity of the system, and hence increase the risk [Kirkpatrick et al. 1992], the cost and the schedule, where development is to occur.
- May result in unforeseen restrictions on operations to meet security or safety requirements.

Where there are specific concerns with the level of integration, it is important that those concerns are included in the specification, even as examples, to provide guidance to the tenderer. Integration requirements also need special attention to ensure that they will not result in unnecessarily restrictive side effects, or unduly affect risk, cost and schedule.

6.9.5 Casualty mode operation

Casualty mode operation refers to the performance of the system in the presence of failures of subsystems or components. It is related to integration in that a high level of integration may enable the reconfiguration of the system to provide high priority functions (by the ability to assign the same function to different consoles, for example). Integration may also impair casualty mode operation, however, if a failure in one subsystem leads to the unavailability of another as a result of their interconnection.

Specifying casualty mode operation can be very difficult because the potential for reconfiguration or independence of operation, and the risks of one failure affecting other components, will depend strongly on the system architecture (which is unknown at the time of specification). In some cases the drafter's only option is to provide a prioritisation of critical functions (see section 6.5) and to indicate that there is a requirement for continued operation, possibly at a reduced level of effectiveness and efficiency, in the event of subsystem or component failure. Evaluation of this requirement would then be based on the different casualty mode approaches proposed by tenderers.

6.9.6 Specialty engineering areas

Both of our surveys identified functional areas where the requirements are difficult to specify and which require specialist skills and advice. The sources of authoritative advice in these areas need to be identified and, where it is not available, studies need to be carried out to identify the key issues and the basis for requirements.

Candidate areas include:

- Reliability, availability and maintainability (RAM).
- Support of operational systems, particularly software upkeep, maintenance and system training.
- Applicability of Government and Defence information system standards.
- Multi-level security requirements.
- Safety critical systems or subsystems.

6.10 Non-functional requirements

Requirements which do not relate directly to the performance or functions of a system are often referred to as non-functional requirements. Typical non-functional requirements are the "ilities" (availability, maintainability, compatibility, portability, modifiability and other quality factors), physical aspects, build standards and user interfaces. It should be noted that, despite their name, these requirements usually do contribute to the overall effectiveness and/or efficiency of a system.

It is universally agreed that non-functional requirements are a serious area of risk. Typically they are difficult to define and respond to, and are often difficult to verify. In some cases, such as requirements for user interfaces (which certainly affect the efficiency of a system), they need subjective verification. All non-functional requirements need to be considered and specified, and should be derived from the operational requirements where possible. The use of a comprehensive checklist by Navy would assist in this regard.

One of the problems in "capturing" non-functional requirements is that many are specifically intended to reduce risks in the design, development and production of the system, or address functions which are dependent on aspects of the design. For example, specifying requirements for maintainability, portability, built-in testing or user interfaces would be much easier if the architecture and user interface philosophy for the system are defined. The risks

inherent in the approach taken would then be more evident. Similarly, the requirements for training and system support would be easier to define after the system has been designed. In addition, because of the emphasis on what the proposed system may consist of, it is too easy to lose sight of the users' needs and focus instead on the design and development activities.

Criticisms of non-functional requirements often stem from the drafters trying to reduce risks in an assumed design (assumed by the drafters, that is). Because the drafters' experience will usually be from previous projects using older technology, and for other reasons, the assumptions will not always be correct, and the constraints they specify may unnecessarily restrict the design of the system.

Other problems stem from ambiguous and unverifiable requirements such as "the system shall be highly reliable" and "the system shall be easily maintainable", which reveal the drafters' inability to define useful requirements in these areas.

It has been stated earlier in this paper that the development of requirements is difficult. We consider that for many non-functional requirements it is impossible without specialist requirements engineering assistance. Considering the risks involved, we recommend that Navy should consider developing these specialist skills in selected staff.

As a general approach to the specification of non-functional requirements, we suggest that the task requires much effort and discipline to restrict requirements to those genuinely needed by the system users, a broad understanding of modern system architectures and development techniques, and the ability to anticipate and reduce risks by the application of appropriately worded requirements.

6.11 Electronic form of specifications

Specifications are now routinely developed using wordprocessors or databases. Tenderers value the provision of specifications in electronic form for a number of reasons, and it is recommended that the practice of supplying this as part of the RFT should continue.

6.11.1 Tools used for specification development

The use of computer based tools for specification development can be a valuable asset in producing a good specification, and can contribute to the consistency, readability and modifiability of the document. Wordprocessors and database tools offer different benefits as follows:

- Databases can contain much more than the requirements. Additional information can include justification for the requirements, traceability links to other specifications, and administrative data, such as the personnel or agencies responsible for the requirement.

- The printing of specifications from databases will guarantee some consistency of layout and style. Attaining the same consistency using a wordprocessor requires a more disciplined approach.
- Wordprocessors offer a much greater flexibility in formatting options, and graphics and tables may be incorporated more easily.
- Wordprocessors offer more sophisticated document preparation functions including proofing tools and the advantages of automatic cross reference update and table of contents generation.
- Automated hierarchical numbering is more difficult in a database.
- It is easier to produce each baseline specification using a wordprocessor, because all of the information may be contained in a single file.

The use of currently available protocols for dynamic linking between tools and information sources, such as OLE and ODBC on PC computers, may make the combined use of a wordprocessor and database a feasible proposition, providing the benefits of both types of tool.

It is important that Navy standardise on the use of a specification development tool, so that comprehensive guidance can be provided on efficient and effective use of the tool, which complements other guidance on requirements engineering practices.

The following considerations should be included in the choice of a tool or tools:

a. **Database or wordprocessor?** Some of the potential advantages and disadvantages are detailed above.

b. **Use of a popular product.** There were concerns about the use of Filemaker Pro expressed to us in both our industry and project surveys. Criticisms included limitations in its functionality and capacity, its lack of compatibility with other more widely used products, and the limited number of staff competent in its use, both within Navy and industry.

c. **Suitability for specification development.** The tool or tools should facilitate in the automation of specification development, contributing to the readability, consistency and ease of modification.

6.11.2 Consistency in the use of tools

If tools are to be used effectively and efficiently within Navy, they should be used consistently. This will facilitate the drafters' task when several are working on the same document, improve the layout and presentation of the specification, and make the specification more amenable to automatic parsing by tenderers. Consistency of use will be facilitated by the common use of styles, cross references, automatic numbering, templates and macros and the provision of training and written guidance in the use of the tools.

Examples include the table of contents and cross references. The table of contents is essential to the navigation and understanding of the specification and should be available and accurate at all stages of the specification development. This can only be achieved effectively and efficiently through

automatic generation. Guidance on the use of the tool should also encompass practices to ensure the table of contents can be so generated.

Cross references must be accurate and updated through changes to the specification. This can only be achieved if the tool supports cross references, the drafters know how to use these features, and all drafters use them identically.

6.11.3 Miscellaneous issues

Our industry survey resulted in several useful recommendations for the use of electronic versions of specifications, which we endorse:

a. **Separation of graphics.** It is preferable that drawings and other graphics within specifications be included as separate files and linked into the specification file. This will usually reduce the size of the individual files making them easier to handle. Many personnel will not need to refer to the graphics (which usually are amplifying data), and will only work with the text file.

b. **Classified material.** Classified documents require special handling. For electronic documents in particular, this can make the use of the documents quite difficult, particularly for tenderers with subcontractors or partners who are geographically remote. Where feasible without strongly affecting the readability of the specification as a whole, it is recommended that classified requirements and information be provided in a separate document. If the classified material consists primarily of performance figures, the reference to these figures by codes in the unclassified body of the specification should be the preferred practice, such that the sense of the specification is unaffected.

   Where this cannot be achieved, the classification of individual clauses should be marked (portion marking). In such a case the provision of a sanitised version of the specification should also be considered.

c. **Configuration management.** The final review of the RFT package should check that there is exact correspondence between the paper and electronic versions of the specifications.

7. Communication with the tenderers

7.1 Operational requirements information

Our survey of Australian suppliers of complex systems showed that all potential tenderers need more information about the operational requirements. They feel that having such information would allow them to significantly improve their ability to respond to the requirements, both in understanding the relative priorities of different functions and in reducing the possibility of misinterpretation.

We strongly endorse this view. It is impossible in practice to completely specify the requirements for a complex system such that a supplier with limited experience of the application domain can build an adequate system in isolation. It is essential that a contractor's tender preparation and the development teams not only have good specifications, but that they also have a good understanding of why the system is needed and how it is likely to be used.

One reason for this is that a contractor needs to make numerous design choices which are not directly covered by the specification, in both the tender preparation and development phases. For various reasons, including both the sheer number of design choices which need to be made and the limited access to Navy staff during tender preparation or development, many of these choices are made without discussion with project staff or users. In the absence of adequate operational knowledge, many of these choices will at best be sub-optimal, and at worst become serious problems, either during the project or in service. The fact that many of these deficiencies may be detected months or even years after the decision is made will also increase the cost of rectification (see section 1.3).

In our experience developers are also more motivated to accept requested changes if they can appreciate the operational value of such changes. This has been shown on several projects where contractor personnel have belatedly been given direct exposure to the operational environment.

We recommend that all projects strive to provide potential tenderers with as much operational information as can reasonably be provided. It is preferable that such information is not contained in the RFT specification, however, because of the tentative nature of some of the information, the bulk of the material required, and the risk of conflicts of precedence. Some or all of the following methods will be useful, where appropriate:

- Direct exposure of tenderers to the operational environment, e.g. sea rides.
- Informal discussion of requirements with potential users of the system and the specification drafters.
- Provision of typical scenario information, including the procedures which are likely to be followed within those scenarios.
- A description of the support philosophies and concepts.
- The use of current systems and a discussion of their deficiencies.

Contractors can assist in this process by suggesting the information that they feel they need.

It is also important that developers have access to consistent and authoritative advice relating to the operational use of the system during the development phase, but this is outside the scope of this study.

7.2 Pre-release of specifications

We also recommend that projects consider the pre-release of draft specifications to potential tenderers. This practice was strongly supported both in our

industry survey and in our survey of Navy projects. It provides tenderers with an early view of the draft requirements and an opportunity to discuss requirements informally with the Project and users. It also serves as an additional test for the quality of the specification.

As Millett [1994] states:

> The Government should conduct in-depth dialogue with potential offerors through draft Requests for Proposal (RFP) and pre-proposal conferences, to ensure that both the requirements of the Government and the range of offeror capabilities are understood. This dialogue, including the user, the procuring activity, and potential offerors, will ensure that the requirement performance is precisely defined.

It is highly unlikely that the discussion of draft specifications will be effective with groups of tenderers. Contractors are notoriously tight-lipped in the presence of their competitors.

For a complex system pre-release should occur at least 10 weeks (preferably longer) prior to the release of the RFT, allowing 4 weeks for contractors to review the specification, 2 weeks for discussions and 4 weeks for analysis, changes and agreement.

7.3 Reduction of conflicting information

In our industry survey some participants stated that they gained a different impression of the priority of some functions in discussions with different Navy personnel. Navy policy states that official communications between Navy projects and potential suppliers are to be via the Project Authority. In our experience this policy is well known both within Navy and industry, and is generally enforced and followed. The fact that informal communications are providing conflicting information is a concern, but is perhaps understandable. Specialists tend to regard their own part of a project as the most important. Any solution to this problem should address the promotion of a firm official position with regard to priorities rather than reduce the already limited dialogue between the tenderers and Navy's users and specialists.

We believe that this problem will be reduced if the potential tenderers are provided with comprehensive information regarding the operation requirement (section 7.1) and a clear prioritisation of requirements (section 6.5).

8. Miscellaneous issues

8.1 Personnel issues

In our survey of current projects, the following skills and experience were identified as being necessary in the specification drafting team. Not surprisingly, these are predominantly systems engineering skills.

- Contractual experience - an understanding of what can go wrong due to inadequate specification.
- Understanding and appreciation of the operational requirements.
- Specification writing experience.
- Competent use of the English language.
- Requirements engineering skills.
- Appreciation of the need for configuration management.
- Experience in risk analysis.
- Experience in validation techniques.
- Ability to analyse and define non-functional requirements.
- Some familiarity with relevant specialty engineering areas, including software, safety, security and quality engineering.

These skills will be rarely be found in a single person in any organisation. It is important, however, that most of them are present in the specification drafting *team*, and that the remainder can be provided on a part time basis, either in the form of consultants to the project or as part of a specification review team.

This type of problem is often solved in the engineering profession by a combination of a good process (see section 9) and a core of expertise (a "nest of owls" [National Research Council 1989]) which can provide the following assistance:

- Direct specialist assistance to projects as consultants.
- Review of specifications.
- Review and improvement of the specification process.

Specification writing is not easy. The language skills can be learned by many people who already have a good command of written English. Fostering the right attitude and providing the essential experience is much more difficult. The ability to draft good specifications needs to be recognised as a specialist skill in itself [Kalms 1990].

We consider that the risks of poor specifications are so high (see section 1.3) that it is preferable to delay a project rather than proceed with a specification drafting team which has less than the needed combination of skills and experience. It should be noted that the risks are even higher in projects based on high level performance based requirements (see section 3.6) and their control will require even higher standards of skill and experience.

8.2 The use of tools

8.2.1 Overview

It was originally intended that this study should investigate and evaluate different requirements engineering tools. This has not been possible due to both time constraints and our inability to obtain evaluation copies of several of the contenders. It is likely however that such an investigation will be carried out in a subsequent study. At this stage a broad overview of tools is provided.

There are numerous tools available for requirements engineering. Typical tools include the following:

- General purpose tools such as wordprocessors, database products and spreadsheets.
- Analysis tools which provide a broad diversity of techniques which may be relevant in specific projects, such as CASE tools, modelling tools and tools which aid in decision making.
- Requirements tracing tools.

The need for tools in the early phase of a project is far less than that required during design and development and, apart from those addressed below, is not critical to the development of good specifications.

The following tools are regarded as important in the development of RFT specifications.

8.2.2 Specification drafting tools

Specifications are typically developed using either a wordprocessor or a database tool, as discussed in section 6.11. The use of such tools in a systematic and competent manner ensures the consistency of the format of the specification and allows changes to be made quickly and simply. The ability to generate tables of contents and cross references automatically aids in the usability and correctness of the specification.

8.2.3 Requirements tracing tools

There are several requirements tracing tools available ranging in price from about $5000 to $50000 per user. The following list summarises those which are more widely known:

- RDT (GEC-Marconi Systems Australia)
- RTM (GEC-Marconi UK)
- RDD-100 (Ascent Logic)
- CORE (Vitech)
- SLATE (TD Technologies)
- DOORS (Zycad)

There is a great deal of variation in the functionality offered by these tools, with some doing much more than simply managing requirements and their links, and providing reports. Most of the tools run on a variety of computers, including PCs. Our industry and project surveys showed that these tracing tools are rarely used either by contractor or Navy personnel in Navy projects, with a tendency instead to use a general purpose database package to track requirements.

This may be due in part to the cost of the tools. To be generally useful, it will be necessary for several users to use the tool, requiring a number of licenses, resulting in a substantial investment. Many potential users apparently see only a limited benefit is using such tools rather than a general purpose database package, and cannot justify the cost. This attitude could change as the practice

of systematic requirements tracing becomes more common, and the benefits of sophisticated tools are better recognised.

It is important that if Navy should decide to use a specific requirements tracing tool, that a common tool be used throughout Navy, and throughout Defence. This will provide obvious efficiencies in training, guidance and the ability of personnel to transfer between projects. It should also reduce the cost of licensing for such tools.

The cost of such tools to Navy or Defence could be considerable. Navy is likely to need at least 50 licences, possibly costing $250,000 or more. A tool could be developed for much less than this amount specifically to Navy's needs (which are almost certainly less than the rich functionality offered by many of the commercial tools). This alternative should be considered if the licensing cost argues against the purchase of such tools.

8.2.4 Decision making tools

Decision making tools can offer reasonable assistance in prioritising requirements. One example is Criterium Decision Plus (Sygenex) which runs on PC computers.

8.3 Australian industry involvement

RFT specifications should not reduce the opportunities for Australian industry involvement (AII) in system development. Although they may have extensive system development experience, Australian companies rarely have the diversity of experience of their off-shore competitors in developing systems for Defence applications. Consequently, they may have more difficulty in proposing or developing systems where the functionality required is based on a high level specification, resulting in a higher level of development risk (see section 3.6).

This risk will be reduced by providing all tenderers with comprehensive information regarding the intended operational use of the system (section 7.1) and by deriving requirements to a more detailed level in areas where Australian companies are assessed as having less experience.

8.4 Alternative acquisition strategies

Most system contracts in the past have followed the "big bang" model, where there is a single design and development phase, following by acceptance of the total system. There are indications that many procurements in the future, particularly for information systems, will consist of several phases, with each successive phase delivering a usable system and adding to the system functionality.

This approach is attractive for several reasons:

- A working (but incomplete) system is delivered much earlier, allowing use of and experience with the system while development proceeds with

the remaining functionality. This will allow specification defects to be detected earlier, and facilitate changes in requirements.

- All complex system developments involve a medium to high level of schedule risk, usually in the software development. By ensuring that critical functions are delivered early in the development, late delivery is likely to have less impact on the overall contract deliverables. Providing the navigation and ship control interface functions in an early phase of a combat system development, for example, may allow early ship trials to proceed, even though the combat system is incomplete.
- Where it is difficult to specify requirements for specialised functions, these may be scheduled for a late phase of the development, allowing additional time and the additional experience in the use of the system to assist in the refinement of requirements.

Such approaches can be categorised as iterative procurement methods, of which *evolutionary acquisition* is the best known.

There is no doubt that the specifications for systems developed using such methods will need to be different from those used for the traditional "big bang" approach. Not only will users need to identify the functionality to be delivered early, but they may also need to specify the interim performance for functions which are only partially provided in a scheduled release. There is also likely to be a greater variation in the level within specifications, with core functionality (to be delivered in the first or second releases) defined at a detailed level, and some subsequent functions at a higher level, to be refined as the project progresses.

Information Technology Division is currently investigating iterative procurement techniques, including the specification needs. Preliminary results of this investigation are likely to be available early in 1996.

9. The specification process

It is now recognised that it is difficult to consistently produce high quality products without a defined systematic development process. Similarly, improvement of product quality is extremely difficult to achieve without regular review of the process, and evaluation of the resultant products, with an aim towards process improvement. This also applies to specification development, where the specification may be considered as the product.

We consider that the variability in quality of Navy specifications identified in our surveys stems directly from deficiencies both in the specification development process and product evaluation of specifications.

We therefore recommend that Navy establishes a specification development and evaluation process based on the findings of this report and the references provided, and ensures that it is followed within Navy. Regular reviews of the process are also needed to ensure that deficiencies in the process are eliminated and changes in Navy and Defence policy and practices are incorporated.

10. Recommendations

The recommendations below are interim recommendations, based on the findings in this report. Final recommendations including more detailed guidance will be provided in Report 4 of this study, consolidating the recommendations in Reports 2 and 3.

It should be noted that in our experience the majority of the problems identified in this study are not restricted to *Navy* specifications and their development. They are equally valid to other parts of Defence. In many cases it may be more appropriate for the problems to be addressed on a Defence wide basis.

Although carried out under a DSTO task endorsed by Navy, this study is essentially an external review of the relevant practices within Navy. In providing these recommendations, no attempt has been made to assess the impact of their implementation on the organisational structure of Navy Materiel Division or related activities within Navy, or the feasibility of their implementation. It is suggested that these should be examined by a further study within Navy, with assistance from DSTO as considered necessary.

10.1 Specification development process and standardisation

The following recommendations apply to the process of specification development:

a. Navy establishes a defined and monitored specification process based on the findings of this report and the references provided (see section 9).

b. The requirements on which specifications are based are systematically, proactively and visibly validated, to ensure that the written requirements reflect the genuine operational needs (see section 4.4).

c. All specifications for major projects are checked for quality by an independent team skilled in requirements engineering, based on the guidance in sections 4, 5 and 6.

d. Navy standardises on a specification development tool (wordprocessor and/or database), and provides guidance on its use (see section 6.11.1).

e. Navy provides a systematic process for the management of requirements including supporting justification, traceability and change control (see section 4).

f. Navy standardises on a requirements management tool, and provides guidance for its use (see section 8.2.3).

10.2 Format and content of specifications

This report provides numerous recommendations on the format and content of specifications. The following are worth special attention in our opinion.

a. Navy specifications continue to be developed using natural language techniques (see section 3).

b. The level of detail in specifications be decided on the basis of risk, and include consideration of the difficulty, novelty and complexity of the application and the experience of potential tenderers (see sections 3.5, 6.3 and 8.3).

c. That the precedence of requirements at different levels, and in different specifications in the same tree, be clearly defined (see section 6.6).

d. That consideration be given to the prioritisation of requirements, possibly for use only within Navy (see section 6.5).

10.3 Considering the tenderers' needs

The following recommendations are based on the tenderers' needs in using specifications:

a. All projects strive to provide potential tenderers with as much operational information as can reasonably be provided (see section 7.1).

b. Projects consider the pre-release of draft specifications to potential tenderers (see section 7.2).

c. All specifications be provided in electronic form to tenderers (see section 6.11).

d. Guidelines be developed relating to the development of specifications in electronic form with regard to the formatting, use of graphics, security classification techniques and configuration management (see sections 6.11.2 and 6.11.3).

10.4 Personnel

The following recommendations are made with regard to the selection and training of personnel for specification development:

a. Navy ensures that personnel involved in specification development have the required blend of skills and experience (see section 8.1).

b. All specification developments are supported by users and user representatives with the appropriate level and breadth of experience (see section 4.3).

10.5 Areas for special treatment

These recommendations apply to areas commonly addressed in Navy specifications which we believe require special treatment by Navy:

a. Navy institutes a comprehensive review process to upgrade or retire relevant standards, and provide guidance on the use of specific standards (see section 6.2.5).

b. Specific separate justification be given for the use of each standard referenced in specifications, including a mandatory review of the currency, applicability and preference over alternative, including civil, standards (see section 6.2.5).

c. Navy reviews the specification of non-functional requirements, providing guidance on their application (see section 6.10).

d. Navy reviews the specification of other areas where specification is usually difficult, including the requirements for concurrency, integration, expression of statistical performance, the use of COTS (commercial off the shelf components) and specialist functional areas (see section 6.9).

11. Conclusions

Inadequate specifications are the single most important factor in problems occurring in the development of complex systems. The penalties will be experienced in project cost and schedule, unacceptably poor or mismatched performance, and in significantly increased through life costs.

There is no doubt that investment in training, staff selection, time and other resources will reap benefits for Navy in its future systems. This report suggests ways in which this investment might be deployed, and provides other information to assist in development of policy and processes for specification development.

12. References

12.1 Useful reading

The following references are useful reading both in the improvement of Navy's specification development process and in the actual development of specifications. They expand upon the information and many of the suggestions and recommendations in this report.

Kalms' [1990] *Guide to Specification Writing* is an excellent handbook providing an introduction to specification practices. While it is aimed mainly at more routine procurements, much of the advice is valuable for the development of any specification. It is strongly preferred to the *Specification Handbook*, which provides very little additional information and is outdated in style and occasionally in content.

Gause and Weinberg's *Exploring Requirements: Quality before Design* [1989] is a highly readable, non-technical guide to capturing and analysing requirements.

Apart from its readability, its great strength is its concentration only on the requirements phase of a project, avoiding the distractions of proceeding towards design and concentrating on the people issues. It is highly recommended as guide to understanding and thinking about what requirements are and how they should be captured.

Alan Davis's *Software Requirements: Objects, Functions and States* [1993a] is a more technical treatment of requirements engineering. Although targeted specifically at software, it is generally applicable to complex systems as a whole, and is particularly strong in its comparison of some of the more popular and proven requirements engineering methodologies. It is also biased towards practical rather than theoretical advice, unlike much of the modern writing in this area. It also contains an excellent annotated bibliography.

Thayer and Dorfman's *System and Software Requirements Engineering* [1990] is a collection of landmark and new articles related to requirements engineering. It also includes a comprehensive list of definitions. It is recommended reading for anyone intending to carry out research in this area.

12.2 Standards and government documents

1988, *Specification Handbook,* Issue 3, Defence Contracting Organisation, Australian Department of Defence, Canberra.

1994, *Costs of tendering industry survey,* Department of Defence, Australian Government Publishing Service, Canberra.

CEPMAN 1992, *The capital equipment procurement manual,* Australian Department of Defence, AL5.

DOD-STD-2167A 1985, *Defence system software development,* Military Standard, US Department of Defense.

FD(Sea) 1993, *Definition of ADF (Naval) capability requirements,* Minute DGFD(Sea) 97/93 of 01 March 1993.

FD(Sea) 1994, *Users guide to the generic Detailed Operational Requirement,* Draft 31 August 1994.

IEEE 830-1984, *IEEE guide to software requirements specifications,* ANSI/IEEE Standard.

MIL-STD-490A 1985, *Specification practices,* Military Standard, US Department of Defense.

MIL-STD-499B 1994, *Systems engineering,* Draft Military Standard, US Department of Defense, May.

12.3 Other references

Andriole, Stephen J. 1990, *Information system design principles for the 90s*, AFCEA International Press, Fairfax, VA.

Black, Harlan (ed.) 1989, *TTCP requirements engineering and rapid prototyping workshop proceedings, November 1989*, CECOM Centre for Software Engineering, Fort Monmouth.

Boehm, Barry W. 1990, "Verifying and validating software requirements and design specifications", *System and software requirements engineering*, Tutorial, IEEE Computer Society Press, California.

Davis, Alan M. 1993a, *Software requirements: objects, functions and states*, PTR Prentice Hall, Englewood Cliffs, New Jersey.

Davis, Richard L. 1992, *C3I requirements and design*, Divisional Paper ITD-92-09, DSTO.

Davis, Richard L. 1993b, *Lessons learnt on the JP2030 Project*, Divisional Paper ITD-93-31, DSTO.

Fairs, K.G. 1992, *Evolutionary acquisition of command and control systems by the Department of Defence*, Project paper ACSC 8101, Department of Computing Science, Australian Defence Force Academy, Canberra.

Frantz, W. Forrest 1992, "Requirements: A practical, tested approach for breakthrough systems", *Proceedings of NCOSE.*

Freedman, Daniel P. and Gerald M. Weinberg 1990, *Handbook of walkthroughs, inspections, and technical reviews - Evaluating programs, projects and products*, Dorset House, New York.

Fuchs, Norbert E. 1992, "Specifications are (preferably) executable", *Software engineering journal*, September.

Golden, Bruce L., Edward A. Wasil and Patrick T. Harker (eds.) 1989, *The analytic hierarchy process - Applications and studies*, Springer-Verlag, Berlin.

Gause, Donald C. and Weinberg, Gerald M. 1989, *Exploring requirements: quality before design*, Dorset House, New York.

Hofmann, Hubert F. 1993, *Requirements engineering - A survey of methods and tools*, Paper no. 93.05, Institute for Informatics, University of Zurich.

Howes, Norman R. 1990, "On using the users' manual as the requirements specification II", *System and software requirements engineering*, Tutorial, IEEE Computer Society Press, California.

Humphrey, Watts S. 1990, *Managing the software process*, Addison-Wesley, Reading, Massachusetts.

Jaffe, Matthew S., Nancy G. Leveson, P.E. Heimdahl and Bonnie E. Melhart 1991, "Software requirements analysis for real-time process control systems", *IEEE transactions on software engineering*, March.

Kalms, Bryan 1990, *Guide to specification writing*, Australian Government Publishing Service, Canberra.

Kirkpatrick, Robert J., Julie A. Walker and Robert Firth 1992, "Software development risk management: an SEI appraisal", *SEI Technical Review*, Pittsburg.

LaSala, Kenneth P. 1994, "Identifying profiling system requirements with Quality Function Deployment", *Proceedings of NCOSE.*

Leveson, Nancy G., M.P.E. Heimdahl, H. Hildreth and J.D. Reese 1994, "Requirements specification for process-control systems". *IEEE transactions on software engineering*, Vol.20 No.9, September.

Lichter, Horst, Matthias Schneider-Hufschmidt and Heinz Züllighoven 1994, "Prototyping in industrial software projects - Bridging the gap between theory and practice", *IEEE transactions on software engineering*, Vol.20, No.11, November.

Manos, Kerry L. 1993, "Strategies for preventing future 'requirements creep'", *Proceedings of NCOSE.*

Mar, Brian W. 1994, "Requirements for development of software requirements", *Proceedings of NCOSE.*

McNaugher, Thomas L. 1989, *New weapons old politics - America's military procurement muddle*, Washington DC: The Brookings Institution.

Millett, Jack 1994, *Guide for the preparation and use of performance specifications*, AMC-P 715-17, U.S. Army Material Command, Alexandria, Virginia, March .

National Research Council 1989, *Adapting software development strategies to modern technology*, National Academy Press, Washington DC.

Rhoads, Dean 1992, *Commercial practices for defense acquisition guidebook*, Defense Systems Management College, Ft. Belvoir, Virginia.

Roman, Gruia-Catalin 1990, "A taxonomy of current issues in requirements engineering", *System and software requirements engineering*, Tutorial, IEEE Computer Society Press, California.

Rushforth, Shelagh, Richard L. Davis and Linsay Gallon 1990, *Review of procurement on behalf of the Procurement Division*, Central Computer and Telecommunications Agency (UK) Study performed by Electronics Facilities Design, EFD/8931/FR01A, January.

Scharer, Laura 1990, "Pinpointing requirements", *System and software requirements engineering*, Tutorial, IEEE Computer Society Press, California.

Thayer, Richard H. and Merlin Dorfman (ed.) 1990, *System and Software Requirements Engineering*, Tutorial, IEEE Computer Society Press, California.

Vonk, Roland 1990, *Prototyping - The effective use of CASE technology*, Prentice Hall, Englewood Cliffs, NJ.

Zave, Pamela 1990, "A comparison of the major approaches to software specification and design", *System and software requirements engineering*, Tutorial, IEEE Computer Society Press, California.

Zucconi, Lin 1992, "Safety and reliability issues in safety-related systems", *Proceedings of NCOSE.*

Blank Pages

DISTRIBUTION

	Number of Copies
DEPARTMENT OF DEFENCE	
Defence Science and Technology Organisation	
Chief Defence Scientist and members of the) DSTO Central Office Executive)	1 shared copy for circulation
Counsellor Defence Science, London	Doc Control Data Sheet Only
Counsellor Defence Science, Washington	Doc Control Data Sheet Only
Senior Defence Scientific Adviser	1
Scientific Adviser POLCOM	1
Assistant Secretary Scientific Analysis	1
Director, Aeronautical and Maritime Research Laboratory	1
Chief Air Operations Division	1
Chief Maritime Operations Division	1
Chief Weapon Systems Division	1
Electronics and Surveillance Research Laboratory	
Chief, Information Technology Division	1
Chiefs of other ESRL Divisions	Doc Control Data Sheet Only
Research Leader Command & Control and Intelligence Systems	1
Research Leader Military Computing Systems	1
Research Leader Command Control and Communications	1
Head Software Engineering Group	1
Head, Command Support Systems Group	1
Head, Intelligence Systems Group	1
Head, C3I Systems Engineering Group	1
Heads of other ITD groups	Doc Control Data Sheet Only
Manager Human Computer Interaction Laboratory	Doc Control Data Sheet Only
Executive Officer, Information Technology Division	Doc Control Data Sheet Only
Publications & Publicity Officer ITD	1
Mr Andrew Gabb, C3ISE Group, ITD	25
Mr Derek Henderson, C3ISE Group, ITD	3
Acquisition and Logistics	
First Assistant Secretary, Defence Materiel	1
Assistant Secretary, Joint Project Management	1
Director, Information Management and Communications Engineering	1
First Assistant Secretary, Capital Equipment Programs	1
Assistant Secretary, Project Policy and Evaluation	1
Director Command, Communications and Computing Systems	1
Director, Project Management Education and Training	1
First Assistant Secretary, Industry Involvement and Contracting	1
Assistant Chief of the Defence Force for Logistics	1

Navy Office	
Navy Scientific Adviser	1
Assistant Chief of Naval Staff - Materiel	1
Director General, Naval Engineering Requirements	1
Director General, Naval Engineering Services	1
Director, Naval Combat Systems Engineering	5
Director, Equipment Projects - Navy	1
Army Office	
Scientific Adviser - Army	1
Assistant Chief of the General Staff - Materiel	1
Air Office	
Air Force Scientific Adviser	1
Assistant Chief of the Air Force - Materiel	1
Budget and Management	
Director, Management Audit (Capital Investment)	1
Libraries and Information Services	
Defence Central Library, Technical Reports Centre	1
Manager, Document Exchange Centre (for retention)	1
National Technical Information Services, United States	2
Defence Research Information Centre, United Kingdom	2
Director, Scientific Information Services, Canada	1
Ministry of Defence, New Zealand	1
National Library of Australia	1
Defence Science and Technology Organisation Salisbury, Research Library	2
Library Defence Signals Directorate, Canberra	1
Australian Government Publishing Service	1
British Library, Document Supply Centre	1
Parliamentary Library of South Australia	1
The State Library of South Australia	1
Spares, DSTO Salisbury Research Library	10

Department of Defence

1. Page Classification
UNCLASSIFIED
2. Privacy Marking/Caveat
N/A

DOCUMENT CONTROL DATA SHEET

3a. AR Number	3b. Establishment No.	3c. Type of Report	4. Task Number
009-312 AR-~~008-311~~	DSTO-TR-0192	TECHNICAL REPORT	NAV93/067

5. Document Date	6. Cost Code	7. Security Classification	8. No. of Pages	84
September 1995	WA846373	[U] [U] [U] Document Title Abstract	9. No. of Refs.	43

10. Title

NAVY SPECIFICATION STUDY: REPORT 3 - REQUIREMENTS AND SPECIFICATIONS

S (Secret C (Confi) R (Rest) U (Unclass)

* For UNCLASSIFIED docs with a secondary distribution LIMITATION, use (L) in document box.

11. Author(s)

ANDREW P. GABB

DEREK E. HENDERSON

12. Downgrading/ Delimiting Instructions

N/A

13a. Corporate Author and Address

Information Technology Division
Electronics and Surveillance Research Laboratory
PO Box 1500 SALISBURY SA 5108

13b. Task Sponsor

NAVY

14. Officer/Position responsible for

Security	SOESRL
Downgrading	CITD
Approval for release	CITD

15. Secondary Release Statement of this Document

APPROVED FOR PUBLIC RELEASE

Any enquiries outside stated limitations should be referred through DSTIC, Defence Information Services, Department of Defence, Anzac Park West, Canberra, ACT 2600.

16a. Deliberate Announcement

NO LIMITATION

16b. Casual Announcement (for citation in other documents)

[X] No Limitation | [ ] Ref. by Author, Doc No and date only

17. DEFTEST Descriptors	18. DISCAT Subject Codes
Royal Australian Navy, Computer Systems, Equipment Specifications	

19. Abstract

This report is one of a series examining the specification of complex computer based systems in the Royal Australian Navy (RAN). The study was carried out under Task NAV 93/067, *Review of Specification and Evaluation Practices*. This report analyses the needs of specifications and makes recommendations for their improvement.